

PHP 4.0 Manuel de Référence.

Table of Contents

1	Préface [Notes en ligne]	1
1.1	A propos de ce manuel [Notes en ligne]	1
2	Preface [Notes en ligne]	2
2.1	Les auteurs [Notes en ligne]	2
2.2	Copyright [Notes en ligne]	2
3	Copyright, distribution, historique [Notes en ligne]	3
4	Installation [Notes en ligne]	4
4.1	Télécharger la dernière version [Notes en ligne]	4
4.2	Installation sous UNIX [Notes en ligne]	4
4.2.1	Installation rapide (Version Module Apache) [Notes en ligne]	4
4.2.2	Module fhttpd [Notes en ligne]	6
4.2.3	Autres serveurs web [Notes en ligne]	6
4.2.4	`/usr/local/src/fhttpd' [Notes en ligne]	6
4.2.5	Options de base de données [Notes en ligne]	6
4.2.6	Compilation [Notes en ligne]	6
4.2.7	Tests [Notes en ligne]	7
4.2.8	Performances [Notes en ligne]	7
4.3	Liste complète des options de configuration [Notes en ligne]	7
4.3.1	Configuration pour le support des bases de données [Notes en ligne]	7
4.3.1.1	install.configure.with-adabas [Notes en ligne]	8
4.3.1.2	install.configure.enable-dba [Notes en ligne]	8
4.3.1.3	install.configure.enable-dbase [Notes en ligne]	8
4.3.1.4	install.configure.with-dbase [Notes en ligne]	8
4.3.1.5	install.configure.with-db2 [Notes en ligne]	8
4.3.1.6	install.configure.with-db3 [Notes en ligne]	8
4.3.1.7	install.configure.with-dbm [Notes en ligne]	9
4.3.1.8	install.configure.with-dbmaker [Notes en ligne]	9
4.3.1.9	install.configure.with-empress [Notes en ligne]	9
4.3.1.10	install.configure.enable-filepro [Notes en ligne]	9
4.3.1.11	install.configure.with-filepro [Notes en ligne]	9
4.3.1.12	install.configure.with-gdbm [Notes en ligne]	9
4.3.1.13	install.configure.with-hyperwave [Notes en ligne]	9
4.3.1.14	install.configure.with-ibm-db2 [Notes en ligne]	10
4.3.1.15	install.configure.with-informix [Notes en ligne]	10
4.3.1.16	install.configure.with-ingres [Notes en ligne]	10
4.3.1.17	install.configure.with-interbase [Notes en ligne]	10
4.3.1.18	install.configure.with-ldap [Notes en ligne]	10
4.3.1.19	install.configure.with-mysql [Notes en ligne]	11
4.3.1.20	install.configure.with-mysql [Notes en ligne]	11
4.3.1.21	install.configure.with-ndbm [Notes en ligne]	11
4.3.1.22	install.configure.with-oci8 [Notes en ligne]	11
4.3.1.23	install.configure.with-oracle [Notes en ligne]	11
4.3.1.24	install.configure.with-pgsql [Notes en ligne]	12
4.3.1.25	install.configure.with-solid [Notes en ligne]	12

Table of Contents

4.3.1.26 install.configure.with-sybase-ct [Notes en ligne]	12
4.3.1.27 install.configure.with-sybase [Notes en ligne]	12
4.3.1.28 install.configure.with-openlink [Notes en ligne]	13
4.3.1.29 install.configure.with-iodbc [Notes en ligne]	13
4.3.1.30 install.configure.with-custom-odbc [Notes en ligne]	13
4.3.1.31 install.configure.disable-unified-odbc [Notes en ligne]	13
4.3.1.32 install.configure.with-unixODBC [Notes en ligne]	14
4.3.1.33 install.configure.with-velocis [Notes en ligne]	14
4.3.2 Ecommerce [Notes en ligne]	14
4.3.2.1 install.configure.with-ccvs [Notes en ligne]	14
4.3.2.2 install.configure.with-mck [Notes en ligne]	14
4.3.2.3 install.configure.with-cybercash [Notes en ligne]	14
4.3.2.4 install.configure.with-pfpro [Notes en ligne]	15
4.3.3 Graphisme [Notes en ligne]	15
4.3.3.1 install.configure.enable-freetype-4bit-antialias-hack [Notes en ligne]	15
4.3.3.2 install.configure.with-gd [Notes en ligne]	15
4.3.3.3 install.configure.without-gd [Notes en ligne]	15
4.3.3.4 install.configure.with-imagick [Notes en ligne]	15
4.3.3.5 install.configure.with-jpeg-dir [Notes en ligne]	15
4.3.3.6 install.configure.with-png-dir [Notes en ligne]	16
4.3.3.7 install.configure.enable-t1lib [Notes en ligne]	16
4.3.3.8 install.configure.with-t1lib [Notes en ligne]	16
4.3.3.9 install.configure.with-tiff-dir [Notes en ligne]	16
4.3.3.10 install.configure.with-ttf [Notes en ligne]	16
4.3.3.11 install.configure.with-xpm-dir [Notes en ligne]	16
4.3.4 Divers [Notes en ligne]	17
4.3.4.1 install.configure.disable-bcmath [Notes en ligne]	17
4.3.4.2 install.configure.disable-display-source [Notes en ligne]	17
4.3.4.3 install.configure.disable-libtool-lock [Notes en ligne]	17
4.3.4.4 install.configure.disable-pear [Notes en ligne]	17
4.3.4.5 install.configure.disable-pic [Notes en ligne]	17
4.3.4.6 install.configure.disable-posix [Notes en ligne]	18
4.3.4.7 install.configure.disable-rpath [Notes en ligne]	18
4.3.4.8 install.configure.disable-session [Notes en ligne]	18
4.3.4.9 install.configure.enable-bcmath [Notes en ligne]	18
4.3.4.10 install.configure.enable-c9x-inline [Notes en ligne]	18
4.3.4.11 install.configure.enable-calendar [Notes en ligne]	18
4.3.4.12 install.configure.enable-debug [Notes en ligne]	19
4.3.4.13 install.configure.enable-debugger [Notes en ligne]	19
4.3.4.14 install.configure.enable-discard-path [Notes en ligne]	19
4.3.4.15 install.configure.enable-dmalloc [Notes en ligne]	19
4.3.4.16 install.configure.enable-exif [Notes en ligne]	19
4.3.4.17 install.configure.enable-experimental-zts [Notes en ligne]	19
4.3.4.18 install.configure.enable-fast-install [Notes en ligne]	19
4.3.4.19 install.configure.enable-force-cgi-redirect [Notes en ligne]	20
4.3.4.20 install.configure.enable-inline-optimization [Notes en ligne]	20
4.3.4.21 install.configure.enable-libgcc [Notes en ligne]	20

Table of Contents

4.3.4.22 install.configure.enable-maintainer-mode [Notes en ligne]	20
4.3.4.23 install.configure.enable-memory-limit [Notes en ligne]	20
4.3.4.24 install.configure.enable-safe-mode [Notes en ligne]	20
4.3.4.25 install.configure.enable-satellite [Notes en ligne]	21
4.3.4.26 install.configure.enable-shared [Notes en ligne]	21
4.3.4.27 install.configure.enable-sigchild [Notes en ligne]	21
4.3.4.28 install.configure.enable-static [Notes en ligne]	21
4.3.4.29 install.configure.enable-sysvsem [Notes en ligne]	21
4.3.4.30 install.configure.enable-sysvshm [Notes en ligne]	21
4.3.4.31 install.configure.enable-trans-sid [Notes en ligne]	21
4.3.4.32 install.configure.with-cdb [Notes en ligne]	22
4.3.4.33 install.configure.with-config-file-path [Notes en ligne]	22
4.3.4.34 install.configure.with-cpdflib [Notes en ligne]	22
4.3.4.35 install.configure.with-esoob [Notes en ligne]	22
4.3.4.36 install.configure.with-exec-dir [Notes en ligne]	22
4.3.4.37 install.configure.with-fdtk [Notes en ligne]	22
4.3.4.38 install.configure.with-gnu-ld [Notes en ligne]	23
4.3.4.39 install.configure.with-icap [Notes en ligne]	23
4.3.4.40 install.configure.with-imap [Notes en ligne]	23
4.3.4.41 install.configure.with-imsp [Notes en ligne]	23
4.3.4.42 install.configure.with-java [Notes en ligne]	23
4.3.4.43 install.configure.with-kerberos [Notes en ligne]	23
4.3.4.44 install.configure.with-mcal [Notes en ligne]	24
4.3.4.45 install.configure.with-mcrypt [Notes en ligne]	24
4.3.4.46 install.configure.with-mhash [Notes en ligne]	24
4.3.4.47 install.configure.with-mm [Notes en ligne]	24
4.3.4.48 install.configure.with-mod_charset [Notes en ligne]	24
4.3.4.49 install.configure.with-pdflib [Notes en ligne]	24
4.3.4.50 install.configure.with-readline [Notes en ligne]	24
4.3.4.51 install.configure.with-regex [Notes en ligne]	25
4.3.4.52 install.configure.with-servlet [Notes en ligne]	25
4.3.4.53 install.configure.with-swf [Notes en ligne]	25
4.3.4.54 install.configure.with-system-regex [Notes en ligne]	25
4.3.4.55 install.configure.with-tsrn-pth [Notes en ligne]	25
4.3.4.56 install.configure.with-tsrn-pthreads [Notes en ligne]	25
4.3.4.57 install.configure.with-x [Notes en ligne]	26
4.3.4.58 install.configure.with-zlib-dir [Notes en ligne]	26
4.3.4.59 install.configure.with-zlib [Notes en ligne]	26
4.3.4.60 install.configure.without-pcre-regex [Notes en ligne]	26
4.3.4.61 install.configure.without-posix [Notes en ligne]	26
4.3.5 Réseau [Notes en ligne]	26
4.3.5.1 install.configure.with-curl [Notes en ligne]	26
4.3.5.2 install.configure.enable-ftp [Notes en ligne]	27
4.3.5.3 install.configure.with-ftp [Notes en ligne]	27
4.3.5.4 install.configure.disable-url-fopen-wrapper [Notes en ligne]	27
4.3.5.5 install.configure.with-mod-dav [Notes en ligne]	27
4.3.5.6 install.configure.with-openssl [Notes en ligne]	27

Table of Contents

4.3.5.7	install.configure.with-snmpp [Notes en ligne]	27
4.3.5.8	install.configure.enable-ucd-snmpp-hack [Notes en ligne]	28
4.3.5.9	install.configure.enable-sockets [Notes en ligne]	28
4.3.5.10	install.configure.with-yaz [Notes en ligne]	28
4.3.5.11	install.configure.enable-yp [Notes en ligne]	28
4.3.5.12	install.configure.with-yp [Notes en ligne]	28
4.3.6	Configuration de PHP [Notes en ligne]	28
4.3.6.1	install.configure.enable-magic-quotes [Notes en ligne]	28
4.3.6.2	install.configure.disable-short-tags [Notes en ligne]	29
4.3.6.3	install.configure.enable-track-vars [Notes en ligne]	29
4.3.7	Serveur [Notes en ligne]	29
4.3.7.1	install.configure.with-aolserver-src [Notes en ligne]	29
4.3.7.2	install.configure.with-aolserver [Notes en ligne]	29
4.3.7.3	install.configure.with-apache [Notes en ligne]	29
4.3.7.4	install.configure.with-apxs [Notes en ligne]	29
4.3.7.5	install.configure.enable-versioning [Notes en ligne]	30
4.3.7.6	install.configure.with-fhttpd [Notes en ligne]	30
4.3.7.7	install.configure.with-nsapi [Notes en ligne]	30
4.3.7.8	install.configure.with-phhttpd [Notes en ligne]	30
4.3.7.9	install.configure.with-pi3web [Notes en ligne]	30
4.3.7.10	install.configure.with-roxen [Notes en ligne]	30
4.3.7.11	install.configure.enable-roxen-zts [Notes en ligne]	31
4.3.7.12	install.configure.with-thhttpd [Notes en ligne]	31
4.3.7.13	install.configure.with-zeus [Notes en ligne]	31
4.3.8	Texte et langue [Notes en ligne]	31
4.3.8.1	install.configure.with-aspell [Notes en ligne]	31
4.3.8.2	install.configure.with-gettext [Notes en ligne]	31
4.3.8.3	install.configure.with-pspell [Notes en ligne]	31
4.3.8.4	install.configure.with-recode [Notes en ligne]	32
4.3.9	XML [Notes en ligne]	32
4.3.9.1	install.configure.with-dom [Notes en ligne]	32
4.3.9.2	install.configure.enable-sablot-errors-descriptive [Notes en ligne]	32
4.3.9.3	install.configure.with-sablot [Notes en ligne]	32
4.3.9.4	install.configure.enable-wddx [Notes en ligne]	32
4.3.9.5	install.configure.disable-xml [Notes en ligne]	32
4.3.9.6	install.configure.with-xml [Notes en ligne]	33
4.4	Installation sous Windows 95/98/NT [Notes en ligne]	33
4.4.1	Installation [Notes en ligne]	33
4.4.2	Windows 95/98/NT et PWS/IIS 3 [Notes en ligne]	34
4.4.3	Windows NT et IIS 4 [Notes en ligne]	35
4.4.4	Windows 9x/NT et Apache 1.3.x [Notes en ligne]	35
4.4.5	Omni HTTPd 2.0b1 pour Windows [Notes en ligne]	35
4.4.6	Modules PHP [Notes en ligne]	36
4.4.7	Installshield [Notes en ligne]	36
4.5	Problèmes? [Notes en ligne]	37
4.5.1	Lisez la FAQ [Notes en ligne]	37
4.5.2	Rapport de bug [Notes en ligne]	37

Table of Contents

4.5.3	Autres problèmes [Notes en ligne]	37
5	Introduction [Notes en ligne]	38
5.1	Qu'est ce que PHP? [Notes en ligne]	38
5.2	What can PHP do? [Notes en ligne]	38
5.3	La g�n�se du PHP [Notes en ligne]	39
6	S�curit� [Notes en ligne]	40
6.1	CGI binary [Notes en ligne]	40
6.1.1	Faiblesses connues [Notes en ligne]	40
6.1.2	Cas 1: Tous les fichiers sont publics [Notes en ligne]	41
6.1.3	Cas 2: Utilisation de la directive de compilation --enable-force-cgi-redirect [Notes en ligne]	41
6.1.4	Cas 3: Utilisation du "doc_root" ou du "user_dir" [Notes en ligne]	41
6.1.5	Cas 4: L'ex�cutable PHP � l'ext�rieur de l'arborescence du serveur [Notes en ligne]	42
6.2	Module Apache [Notes en ligne]	42
7	Configuration [Notes en ligne]	43
7.1	Le fichier de configuration [Notes en ligne]	43
7.1.1	Directives de configuration g�n�rale [Notes en ligne]	43
7.1.1.1	ini.asp-tags [Notes en ligne]	43
7.1.1.2	ini.auto-append-file [Notes en ligne]	44
7.1.1.3	ini.auto-prepend-file [Notes en ligne]	44
7.1.1.4	ini.cgi-ext [Notes en ligne]	44
7.1.1.5	ini.display-errors [Notes en ligne]	44
7.1.1.6	ini.doc-root [Notes en ligne]	44
7.1.1.7	ini.engine [Notes en ligne]	44
7.1.1.8	ini.error-log [Notes en ligne]	45
7.1.1.9	ini.error-reporting [Notes en ligne]	45
7.1.1.10	ini.open-basedir [Notes en ligne]	45
7.1.1.11	ini.gpc-order [Notes en ligne]	45
7.1.1.12	ini.ignore-user-abort [Notes en ligne]	46
7.1.1.13	ini.include-path [Notes en ligne]	46
7.1.1.14	ini.isapi-ext [Notes en ligne]	46
7.1.1.15	ini.log-errors [Notes en ligne]	46
7.1.1.16	ini.magic-quotes-gpc [Notes en ligne]	46
7.1.1.17	ini.magic-quotes-runtime [Notes en ligne]	47
7.1.1.18	ini.magic-quotes-sybase [Notes en ligne]	47
7.1.1.19	ini.max-execution-time [Notes en ligne]	47
7.1.1.20	ini.memory-limit [Notes en ligne]	47
7.1.1.21	ini.nsapi-ext [Notes en ligne]	47
7.1.1.22	ini.short-open-tag [Notes en ligne]	47
7.1.1.23	ini.sql.safe-mode [Notes en ligne]	48
7.1.1.24	ini.track-errors [Notes en ligne]	48
7.1.1.25	ini.track-vars [Notes en ligne]	48
7.1.1.26	ini.upload-tmp-dir [Notes en ligne]	48
7.1.1.27	ini.user-dir [Notes en ligne]	48
7.1.1.28	ini.warn-plus-overloading [Notes en ligne]	48

Table of Contents

7.1.2 Configuration des directives concernant le mail [Notes en ligne]	49
7.1.2.1 ini.smtp [Notes en ligne]	49
7.1.2.2 ini.sendmail-from [Notes en ligne]	49
7.1.2.3 ini.sendmail-path [Notes en ligne]	49
7.1.3 Directives de configuration du "Safe Mode" [Notes en ligne]	49
7.1.3.1 ini.safe-mode [Notes en ligne]	49
7.1.3.2 ini.safe-mode-exec-dir [Notes en ligne]	49
7.1.4 Directives de configuration de débbugage. [Notes en ligne]	50
7.1.4.1 ini.debugger.host [Notes en ligne]	50
7.1.4.2 ini.debugger.port [Notes en ligne]	50
7.1.4.3 ini.debugger.enabled [Notes en ligne]	50
7.1.5 Directives de chargement des extensions [Notes en ligne]	50
7.1.5.1 ini.enable-dl [Notes en ligne]	50
7.1.5.2 ini.extension-dir [Notes en ligne]	50
7.1.5.3 ini.extension [Notes en ligne]	51
7.1.6 MySQL Configuration Directives [Notes en ligne]	51
7.1.6.1 ini.mysql.allow-persistent [Notes en ligne]	51
7.1.6.2 ini.mysql.default-host [Notes en ligne]	51
7.1.6.3 ini.mysql.default-user [Notes en ligne]	51
7.1.6.4 ini.mysql.default-password [Notes en ligne]	51
7.1.6.5 ini.mysql.max-persistent [Notes en ligne]	51
7.1.6.6 ini.mysql.max-links [Notes en ligne]	52
7.1.7 Directives de configuration mSQL [Notes en ligne]	52
7.1.7.1 ini.msql.allow-persistent [Notes en ligne]	52
7.1.7.2 ini.msql.max-persistent [Notes en ligne]	52
7.1.7.3 ini.msql.max-links [Notes en ligne]	52
7.1.8 Directives de configuration Postgres [Notes en ligne]	52
7.1.8.1 ini.pgsq.allow-persistent [Notes en ligne]	52
7.1.8.2 ini.pgsq.max-persistent [Notes en ligne]	52
7.1.8.3 ini.pgsq.max-links [Notes en ligne]	53
7.1.9 Directives de configuration Sybase [Notes en ligne]	53
7.1.9.1 ini.sybase.allow-persistent [Notes en ligne]	53
7.1.9.2 ini.sybase.max-persistent [Notes en ligne]	53
7.1.9.3 ini.sybase.max-links [Notes en ligne]	53
7.1.10 Sybase-CT Configuration Directives [Notes en ligne]	53
7.1.10.1 ini.sybct.allow-persistent [Notes en ligne]	53
7.1.10.2 ini.sybct.max-persistent [Notes en ligne]	53
7.1.10.3 ini.sybct.max-links [Notes en ligne]	54
7.1.10.4 ini.sybct.min-server-severity [Notes en ligne]	54
7.1.10.5 ini.sybct.min-client-severity [Notes en ligne]	54
7.1.10.6 ini.sybct.login-timeout [Notes en ligne]	54
7.1.10.7 ini.sybct.timeout [Notes en ligne]	54
7.1.10.8 ini.sybct.hostname [Notes en ligne]	55
7.1.11 Directives de configuration Informix [Notes en ligne]	55
7.1.11.1 ini.ifx.allow-persistent [Notes en ligne]	55
7.1.11.2 ini.ifx.max-persistent [Notes en ligne]	55
7.1.11.3 ini.ifx.max-links [Notes en ligne]	55

Table of Contents

7.1.11.4	ini.ifx.default-host [Notes en ligne]	55
7.1.11.5	ini.ifx.default-user [Notes en ligne]	55
7.1.11.6	ini.ifx.default-password [Notes en ligne]	56
7.1.11.7	ini.ifx.blobinfile [Notes en ligne]	56
7.1.11.8	ini.ifx.textasvarchar [Notes en ligne]	56
7.1.11.9	ini.ifx.byteasvarchar [Notes en ligne]	56
7.1.11.10	ini.ifx.charasvarchar [Notes en ligne]	56
7.1.11.11	ini.ifx.nullformat [Notes en ligne]	56
7.1.12	Directives de configuration pour les calculs mathématiques. [Notes en ligne]	57
7.1.12.1	ini.bcmath.scale [Notes en ligne]	57
7.1.13	Directives de configuration du navigateur. [Notes en ligne]	57
7.1.13.1	ini.browscap [Notes en ligne]	57
7.1.14	Directives de configuration du driver ODBC unifié [Notes en ligne]	57
7.1.14.1	ini.uodbc.default-db [Notes en ligne]	57
7.1.14.2	ini.uodbc.default-user [Notes en ligne]	57
7.1.14.3	ini.uodbc.default-pw [Notes en ligne]	57
7.1.14.4	ini.uodbc.allow-persistent [Notes en ligne]	57
7.1.14.5	ini.uodbc.max-persistent [Notes en ligne]	58
7.1.14.6	ini.uodbc.max-links [Notes en ligne]	58
8	Caractéristiques [Notes en ligne]	59
8.1	Gestion des connexions [Notes en ligne]	59
8.2	Cookies [Notes en ligne]	60
8.3	Gestion des erreurs [Notes en ligne]	60
8.4	Gestion des chargements de fichier [Notes en ligne]	60
8.4.1	Chargements de fichiers par méthode POST [Notes en ligne]	60
8.4.2	Erreurs classiques [Notes en ligne]	61
8.4.3	Chargement multiples de fichiers [Notes en ligne]	61
8.4.4	Chargement par méthode PUT [Notes en ligne]	62
8.5	Authentification HTTP avec PHP [Notes en ligne]	63
8.6	Création d'images [Notes en ligne]	64
8.7	Connexions persistantes aux bases de données [Notes en ligne]	65
8.8	Utilisation des fichiers à distance [Notes en ligne]	65
9	Langage [Notes en ligne]	67
9.1	La syntaxe de base [Notes en ligne]	67
9.1.1	Le passage du HTML au PHP [Notes en ligne]	67
9.1.2	Le séparateur d'instruction [Notes en ligne]	67
9.1.3	Comments [Notes en ligne]	68
9.2	Les constantes [Notes en ligne]	68
9.3	Les structures de contrôle [Notes en ligne]	69
9.3.1	if [Notes en ligne]	70
9.3.2	else [Notes en ligne]	70
9.3.3	elseif [Notes en ligne]	71
9.3.4	Syntaxe alternative [Notes en ligne]	71
9.3.5	while [Notes en ligne]	72
9.3.6	do..while [Notes en ligne]	72

Table of Contents

9.3.7	for [Notes en ligne]	73
9.3.8	foreach [Notes en ligne]	74
9.3.9	break [Notes en ligne]	75
9.3.10	continue [Notes en ligne]	76
9.3.11	switch [Notes en ligne]	76
9.3.12	require() [Notes en ligne] [Exemples]	78
9.3.13	include() [Notes en ligne] [Exemples]	79
9.4	Les expressions [Notes en ligne]	81
9.5	Fonctions [Notes en ligne]	83
9.5.1	Les fonctions utilisateurs [Notes en ligne] [Exemples]	83
9.5.2	Les arguments de fonction [Notes en ligne] [Exemples]	84
9.5.2.1	Passage d'arguments par référence [Notes en ligne] [Exemples]	84
9.5.2.2	Valeur par défaut des arguments [Notes en ligne] [Exemples]	85
9.5.2.3	Variable-length argument lists [Notes en ligne] [Exemples]	85
9.5.3	Les valeurs de retour [Notes en ligne] [Exemples]	86
9.5.4	old function [Notes en ligne] [Exemples]	86
9.5.5	Variable fonctions [Notes en ligne] [Exemples]	86
9.6	Classes et objets [Notes en ligne]	87
9.6.1	class [Notes en ligne]	87
9.7	Les opérateurs [Notes en ligne]	88
9.7.1	Les opérateurs arithmétiques [Notes en ligne]	89
9.7.2	Les opérateurs d'assignement [Notes en ligne]	89
9.7.3	Bitwise Operators [Notes en ligne]	89
9.7.4	Opérateurs de comparaison [Notes en ligne]	90
9.7.5	Opérateur de contrôle d'erreur [Notes en ligne]	90
9.7.6	Opérateur d'exécutions [Notes en ligne]	91
9.7.7	Incrementing/Decrementing Operators [Notes en ligne]	91
9.7.8	Les opérateurs logiques [Notes en ligne]	92
9.7.9	La précedence des opérateurs [Notes en ligne]	92
9.7.10	String Operators [Notes en ligne]	93
9.8	Types [Notes en ligne]	93
9.8.1	Entiers [Notes en ligne]	94
9.8.2	Les nombres à virgule flottante [Notes en ligne]	94
9.8.3	Les chaînes de caractères [Notes en ligne]	94
9.8.3.1	Conversion de type [Notes en ligne]	95
9.8.4	Les tableaux [Notes en ligne]	96
9.8.4.1	Tableaux à une dimension [Notes en ligne]	96
9.8.4.2	Tableaux à plusieurs dimensions [Notes en ligne]	97
9.8.5	Les objets [Notes en ligne]	98
9.8.5.1	Initialisation d'un objet [Notes en ligne]	98
9.8.6	Définition du type [Notes en ligne]	98
9.8.6.1	Transtypage [Notes en ligne]	99
9.9	Les variables [Notes en ligne]	100
9.9.1	Essentiel [Notes en ligne]	100
9.9.2	Variables prédéfinies [Notes en ligne]	101
9.9.2.1	Variables Apache [Notes en ligne]	101
9.9.2.2	Variables d'environnement [Notes en ligne]	103

Table of Contents

9.9.2.3	Variables PHP [Notes en ligne]	103
9.9.3	Portée des variables [Notes en ligne]	103
9.9.4	Les variables dynamiques [Notes en ligne]	105
9.9.5	Variables externes à PHP [Notes en ligne]	106
9.9.5.1	Formulaires HTML (GET et POST) [Notes en ligne]	106
9.9.5.2	HTTP Cookies [Notes en ligne]	107
9.9.5.3	Variables d'environnement [Notes en ligne]	107
9.9.5.4	Cas des points dans les noms de variables [Notes en ligne]	107
9.9.5.5	Détermination du type des variables [Notes en ligne]	108
9.10	References Explained [Notes en ligne]	108
9.10.1	What are References [Notes en ligne]	108
9.10.2	What do References [Notes en ligne]	108
9.10.3	What aren't References [Notes en ligne]	109
9.10.4	Returning References [Notes en ligne]	109
9.10.5	Unsetting References [Notes en ligne]	109
9.10.6	Spotting the Reference [Notes en ligne]	110
9.10.6.1	global References [Notes en ligne]	110
9.10.6.2	\$this [Notes en ligne]	110
10	Fonctions [Notes en ligne] [Exemples]	111
10.1	Apache [Notes en ligne]	111
10.1.1	apache_lookup_uri [Notes en ligne] [Exemples]	111
10.1.2	apache_note [Notes en ligne] [Exemples]	111
10.1.3	getallheaders [Notes en ligne] [Exemples]	112
10.1.4	virtual [Notes en ligne] [Exemples]	112
10.1.5	asci2ebcdic [Notes en ligne] [Exemples]	112
10.1.6	ebcdic2ascii [Notes en ligne] [Exemples]	113
10.2	Tableaux [Notes en ligne]	113
10.2.1	array [Notes en ligne] [Exemples]	113
10.2.2	array_count_values [Notes en ligne] [Exemples]	113
10.2.3	array_diff [Notes en ligne] [Exemples]	114
10.2.4	array_flip [Notes en ligne] [Exemples]	114
10.2.5	array_intersect [Notes en ligne] [Exemples]	114
10.2.6	array_keys [Notes en ligne] [Exemples]	115
10.2.7	array_merge [Notes en ligne] [Exemples]	116
10.2.8	array_merge_recursive [Notes en ligne] [Exemples]	116
10.2.9	array_multisort [Notes en ligne] [Exemples]	116
10.2.10	array_pad [Notes en ligne] [Exemples]	117
10.2.11	array_pop [Notes en ligne] [Exemples]	118
10.2.12	array_push [Notes en ligne] [Exemples]	118
10.2.13	array_reverse [Notes en ligne] [Exemples]	119
10.2.14	array_rand [Notes en ligne] [Exemples]	119
10.2.15	array_shift [Notes en ligne] [Exemples]	120
10.2.16	array_slice [Notes en ligne] [Exemples]	120
10.2.17	array_splice [Notes en ligne] [Exemples]	120
10.2.18	array_unique [Notes en ligne] [Exemples]	121
10.2.19	array_unshift [Notes en ligne] [Exemples]	122

Table of Contents

10.2.20 array_values [Notes en ligne] [Exemples]	122
10.2.21 array_walk [Notes en ligne] [Exemples]	123
10.2.22 arsort [Notes en ligne] [Exemples]	123
10.2.23 asort [Notes en ligne] [Exemples]	124
10.2.24 compact [Notes en ligne] [Exemples]	124
10.2.25 count [Notes en ligne] [Exemples]	125
10.2.26 current [Notes en ligne] [Exemples]	125
10.2.27 each [Notes en ligne] [Exemples]	126
10.2.28 end [Notes en ligne] [Exemples]	127
10.2.29 extract [Notes en ligne] [Exemples]	127
10.2.30 in_array [Notes en ligne] [Exemples]	128
10.2.31 key [Notes en ligne] [Exemples]	128
10.2.32 krsort [Notes en ligne] [Exemples]	128
10.2.33 ksort [Notes en ligne] [Exemples]	129
10.2.34 list [Notes en ligne] [Exemples]	129
10.2.35 natsort [Notes en ligne] [Exemples]	130
10.2.36 natcasesort [Notes en ligne] [Exemples]	130
10.2.37 next [Notes en ligne] [Exemples]	131
10.2.38 pos [Notes en ligne] [Exemples]	131
10.2.39 prev [Notes en ligne] [Exemples]	131
10.2.40 range [Notes en ligne] [Exemples]	131
10.2.41 reset [Notes en ligne] [Exemples]	132
10.2.42 rsort [Notes en ligne] [Exemples]	132
10.2.43 shuffle [Notes en ligne] [Exemples]	132
10.2.44 sizeof [Notes en ligne] [Exemples]	132
10.2.45 sort [Notes en ligne] [Exemples]	133
10.2.46 uasort [Notes en ligne] [Exemples]	133
10.2.47 uksort [Notes en ligne] [Exemples]	133
10.2.48 usort [Notes en ligne] [Exemples]	134
10.3 Fonctions Aspell [Notes en ligne]	134
10.3.1 aspell_new [Notes en ligne] [Exemples]	134
10.3.2 aspell_check [Notes en ligne] [Exemples]	135
10.3.3 aspell_check-raw [Notes en ligne] [Exemples]	135
10.3.4 aspell_suggest [Notes en ligne] [Exemples]	136
10.4 Fonctions mathématiques sur des nombres de taille arbitraire [Notes en ligne]	136
10.4.1 bcadd [Notes en ligne] [Exemples]	136
10.4.2 bccomp [Notes en ligne] [Exemples]	136
10.4.3 bcddiv [Notes en ligne] [Exemples]	137
10.4.4 bcmath [Notes en ligne] [Exemples]	137
10.4.5 bcmul [Notes en ligne] [Exemples]	137
10.4.6 bcpow [Notes en ligne] [Exemples]	137
10.4.7 bcscale [Notes en ligne] [Exemples]	137
10.4.8 bcsqrt [Notes en ligne] [Exemples]	137
10.4.9 bcsqrt [Notes en ligne] [Exemples]	137
10.4.9 bcsub [Notes en ligne] [Exemples]	138
10.5 Fonctions de calendrier [Notes en ligne]	138
10.5.1 JDToGregorian [Notes en ligne] [Exemples]	138
10.5.2 GregorianToJD [Notes en ligne] [Exemples]	138

Table of Contents

10.5.3 JDTToJulian [Notes en ligne] [Exemples]	139
10.5.4 JulianToJD [Notes en ligne] [Exemples]	139
10.5.5 JDTToJewish [Notes en ligne] [Exemples]	139
10.5.6 JewishToJD [Notes en ligne] [Exemples]	139
10.5.7 JDTToFrench [Notes en ligne] [Exemples]	140
10.5.8 FrenchToJD [Notes en ligne] [Exemples]	140
10.5.9 JDMonthName [Notes en ligne] [Exemples]	140
10.5.10 JDDayOfWeek [Notes en ligne] [Exemples]	140
10.5.11 easter_date [Notes en ligne] [Exemples]	141
10.5.12 easter_days [Notes en ligne] [Exemples]	141
10.6 CCVS API Functions [Notes en ligne]	142
10.7 Fonctions Classe/Objet [Notes en ligne]	142
10.7.1 get_class_methods [Notes en ligne] [Exemples]	142
10.7.2 get_class_vars [Notes en ligne] [Exemples]	142
10.7.3 get_object_vars [Notes en ligne] [Exemples]	143
10.7.4 method_exists [Notes en ligne] [Exemples]	143
10.8 Support COM pour Windows [Notes en ligne]	143
10.8.1 com_load [Notes en ligne] [Exemples]	143
10.8.2 com_invoke [Notes en ligne] [Exemples]	143
10.8.3 com_propget [Notes en ligne] [Exemples]	143
10.8.4 com_get [Notes en ligne] [Exemples]	144
10.8.5 com_propput [Notes en ligne] [Exemples]	144
10.8.6 com_propset [Notes en ligne] [Exemples]	144
10.8.7 com_set [Notes en ligne] [Exemples]	144
10.9 Fonctions ClibPDF [Notes en ligne]	144
10.9.1 pdf_global_set_document_limits [Notes en ligne] [Exemples]	146
10.9.2 pdf_set_creator [Notes en ligne] [Exemples]	147
10.9.3 pdf_set_title [Notes en ligne] [Exemples]	147
10.9.4 pdf_set_subject [Notes en ligne] [Exemples]	147
10.9.5 pdf_set_keywords [Notes en ligne] [Exemples]	147
10.9.6 pdf_open [Notes en ligne] [Exemples]	147
10.9.7 pdf_close [Notes en ligne] [Exemples]	148
10.9.8 pdf_page_init [Notes en ligne] [Exemples]	148
10.9.9 pdf_finalize_page [Notes en ligne] [Exemples]	148
10.9.10 pdf_finalize [Notes en ligne] [Exemples]	148
10.9.11 pdf_output_buffer [Notes en ligne] [Exemples]	148
10.9.12 pdf_save_to_file [Notes en ligne] [Exemples]	149
10.9.13 pdf_set_current_page [Notes en ligne] [Exemples]	149
10.9.14 pdf_begin_text [Notes en ligne] [Exemples]	149
10.9.15 pdf_end_text [Notes en ligne] [Exemples]	149
10.9.16 pdf_show [Notes en ligne] [Exemples]	150
10.9.17 pdf_show_xy [Notes en ligne] [Exemples]	150
10.9.18 pdf_text [Notes en ligne] [Exemples]	150
10.9.19 pdf_set_font [Notes en ligne] [Exemples]	150
10.9.20 pdf_set_leading [Notes en ligne] [Exemples]	151
10.9.21 pdf_set_text_rendering [Notes en ligne] [Exemples]	151
10.9.22 pdf_set_horiz_scaling [Notes en ligne] [Exemples]	151

Table of Contents

10.9.23 cpdf_set_text_rise [Notes en ligne] [Exemples]	151
10.9.24 cpdf_set_text_matrix [Notes en ligne] [Exemples]	151
10.9.25 cpdf_set_text_pos [Notes en ligne] [Exemples]	151
10.9.26 cpdf_set_char_spacing [Notes en ligne] [Exemples]	152
10.9.27 cpdf_set_word_spacing [Notes en ligne] [Exemples]	152
10.9.28 cpdf_continue_text [Notes en ligne] [Exemples]	152
10.9.29 cpdf_stringwidth [Notes en ligne] [Exemples]	152
10.9.30 cpdf_save [Notes en ligne] [Exemples]	152
10.9.31 cpdf_restore [Notes en ligne] [Exemples]	153
10.9.32 cpdf_translate [Notes en ligne] [Exemples]	153
10.9.33 cpdf_scale [Notes en ligne] [Exemples]	153
10.9.34 cpdf_rotate [Notes en ligne] [Exemples]	153
10.9.35 cpdf_setflat [Notes en ligne] [Exemples]	153
10.9.36 cpdf_setlinejoin [Notes en ligne] [Exemples]	154
10.9.37 cpdf_setlinecap [Notes en ligne] [Exemples]	154
10.9.38 cpdf_setmiterlimit [Notes en ligne] [Exemples]	154
10.9.39 cpdf_setlinewidth [Notes en ligne] [Exemples]	154
10.9.40 cpdf_setdash [Notes en ligne] [Exemples]	154
10.9.41 cpdf_moveto [Notes en ligne] [Exemples]	154
10.9.42 cpdf_rmoveto [Notes en ligne] [Exemples]	155
10.9.43 cpdf_curveto [Notes en ligne] [Exemples]	155
10.9.44 cpdf_lineto [Notes en ligne] [Exemples]	155
10.9.45 cpdf_rlineto [Notes en ligne] [Exemples]	155
10.9.46 cpdf_circle [Notes en ligne] [Exemples]	155
10.9.47 cpdf_arc [Notes en ligne] [Exemples]	156
10.9.48 cpdf_rect [Notes en ligne] [Exemples]	156
10.9.49 cpdf_closepath [Notes en ligne] [Exemples]	156
10.9.50 cpdf_stroke [Notes en ligne] [Exemples]	156
10.9.51 cpdf_closepath_stroke [Notes en ligne] [Exemples]	157
10.9.52 cpdf_fill [Notes en ligne] [Exemples]	157
10.9.53 cpdf_fill_stroke [Notes en ligne] [Exemples]	157
10.9.54 cpdf_closepath_fill_stroke [Notes en ligne] [Exemples]	157
10.9.55 cpdf_clip [Notes en ligne] [Exemples]	157
10.9.56 cpdf_setgray_fill [Notes en ligne] [Exemples]	157
10.9.57 cpdf_setgray_stroke [Notes en ligne] [Exemples]	158
10.9.58 cpdf_setgray [Notes en ligne] [Exemples]	158
10.9.59 cpdf_setrgbcolor_fill [Notes en ligne] [Exemples]	158
10.9.60 cpdf_setrgbcolor_stroke [Notes en ligne] [Exemples]	158
10.9.61 cpdf_setrgbcolor [Notes en ligne] [Exemples]	158
10.9.62 cpdf_add_outline [Notes en ligne] [Exemples]	159
10.9.63 cpdf_set_page_animation [Notes en ligne] [Exemples]	159
10.9.64 cpdf_import_jpeg [Notes en ligne] [Exemples]	159
10.9.65 cpdf_place_inline_image [Notes en ligne] [Exemples]	160
10.9.66 cpdf_add_annotation [Notes en ligne] [Exemples]	160
10.10 CURL, Client URL Library Functions [Notes en ligne]	160
10.10.1 curl_init [Notes en ligne] [Exemples]	161
10.10.2 curl_setopt [Notes en ligne] [Exemples]	161

Table of Contents

10.10.3	curl_exec [Notes en ligne] [Exemples]	163
10.10.4	curl_close [Notes en ligne] [Exemples]	164
10.10.5	curl_version [Notes en ligne] [Exemples]	164
10.11	Fonctions de paiement Cybercash [Notes en ligne]	164
10.11.1	cybercash_encr [Notes en ligne] [Exemples]	164
10.11.2	cybercash_decr [Notes en ligne] [Exemples]	164
10.11.3	cybercash_base64_encode [Notes en ligne] [Exemples]	164
10.11.4	cybercash_base64_decode [Notes en ligne] [Exemples]	165
10.12	Fonctions de dates et heures [Notes en ligne]	165
10.12.1	checkdate [Notes en ligne] [Exemples]	165
10.12.2	date [Notes en ligne] [Exemples]	165
10.12.3	getdate [Notes en ligne] [Exemples]	166
10.12.4	gettimeofday [Notes en ligne] [Exemples]	167
10.12.5	gmmdate [Notes en ligne] [Exemples]	167
10.12.6	gmmktime [Notes en ligne] [Exemples]	167
10.12.7	gmstrftime [Notes en ligne] [Exemples]	167
10.12.8	microtime [Notes en ligne] [Exemples]	168
10.12.9	mktime [Notes en ligne] [Exemples]	168
10.12.10	strftime [Notes en ligne] [Exemples]	169
10.12.11	time [Notes en ligne] [Exemples]	170
10.13	Fonctions dbm [Notes en ligne]	170
10.13.1	dba_close [Notes en ligne] [Exemples]	171
10.13.2	dba_delete [Notes en ligne] [Exemples]	171
10.13.3	dba_exists [Notes en ligne] [Exemples]	172
10.13.4	dba_fetch [Notes en ligne] [Exemples]	172
10.13.5	dba_firstkey [Notes en ligne] [Exemples]	172
10.13.6	dba_insert [Notes en ligne] [Exemples]	172
10.13.7	dba_nextkey [Notes en ligne] [Exemples]	173
10.13.8	dba_popen [Notes en ligne] [Exemples]	173
10.13.9	dba_open [Notes en ligne] [Exemples]	173
10.13.10	dba_optimize [Notes en ligne] [Exemples]	173
10.13.11	dba_replace [Notes en ligne] [Exemples]	174
10.13.12	dba_sync [Notes en ligne] [Exemples]	174
10.14	Fonctions dBase [Notes en ligne]	174
10.14.1	dbase_create [Notes en ligne] [Exemples]	175
10.14.2	dbase_open [Notes en ligne] [Exemples]	175
10.14.3	dbase_close [Notes en ligne] [Exemples]	176
10.14.4	dbase_pack [Notes en ligne] [Exemples]	176
10.14.5	dbase_add_record [Notes en ligne] [Exemples]	176
10.14.6	dbase_replace_record [Notes en ligne] [Exemples]	176
10.14.7	dbase_delete_record [Notes en ligne] [Exemples]	176
10.14.8	dbase_get_record [Notes en ligne] [Exemples]	177
10.14.9	dbase_get_record_with_names [Notes en ligne] [Exemples]	177
10.14.10	dbase_numfields [Notes en ligne] [Exemples]	177
10.14.11	dbase_numrecords [Notes en ligne] [Exemples]	177
10.15	Fonctions dbm [Notes en ligne]	178
10.15.1	dbmopen [Notes en ligne] [Exemples]	178

Table of Contents

10.15.2	dbmclose [Notes en ligne] [Exemples]	178
10.15.3	dbmexists [Notes en ligne] [Exemples]	179
10.15.4	dbmfetch [Notes en ligne] [Exemples]	179
10.15.5	dbminsert [Notes en ligne] [Exemples]	179
10.15.6	dbmreplace [Notes en ligne] [Exemples]	179
10.15.7	dbmdelete [Notes en ligne] [Exemples]	179
10.15.8	dbmfirstkey [Notes en ligne] [Exemples]	179
10.15.9	dbmnextkey [Notes en ligne] [Exemples]	180
10.15.10	dblist [Notes en ligne] [Exemples]	180
10.16	Accès aux dossiers [Notes en ligne]	180
10.16.1	chdir [Notes en ligne] [Exemples]	180
10.16.2	dir [Notes en ligne] [Exemples]	180
10.16.3	closedir [Notes en ligne] [Exemples]	181
10.16.4	opendir [Notes en ligne] [Exemples]	181
10.16.5	readdir [Notes en ligne] [Exemples]	181
10.16.6	rewinddir [Notes en ligne] [Exemples]	182
10.17	Chargement dynamique de fonctions [Notes en ligne]	182
10.17.1	dl [Notes en ligne] [Exemples]	182
10.18	Fonctions DOM XML [Notes en ligne]	182
10.18.1	xmldoc [Notes en ligne] [Exemples]	183
10.18.2	xmldocfile [Notes en ligne] [Exemples]	183
10.18.3	xmltree [Notes en ligne] [Exemples]	183
10.19	Error Handling and Logging Functions [Notes en ligne]	183
10.19.1	error_log [Notes en ligne] [Exemples]	184
10.19.2	error_reporting [Notes en ligne] [Exemples]	184
10.19.3	restore_error_handler [Notes en ligne] [Exemples]	185
10.19.4	set_error_handler [Notes en ligne] [Exemples]	185
10.19.5	trigger_error [Notes en ligne] [Exemples]	187
10.19.6	user_error [Notes en ligne] [Exemples]	187
10.20	Fonction d'exécution de programmes [Notes en ligne]	188
10.20.1	escapeshellcmd [Notes en ligne] [Exemples]	188
10.20.2	exec [Notes en ligne] [Exemples]	188
10.20.3	passthru [Notes en ligne] [Exemples]	188
10.20.4	system [Notes en ligne] [Exemples]	189
10.21	Forms Data Format [Notes en ligne]	189
10.21.1	fdf_open [Notes en ligne] [Exemples]	190
10.21.2	fdf_close [Notes en ligne] [Exemples]	191
10.21.3	fdf_create [Notes en ligne] [Exemples]	191
10.21.4	fdf_save [Notes en ligne] [Exemples]	191
10.21.5	fdf_get_value [Notes en ligne] [Exemples]	191
10.21.6	fdf_set_value [Notes en ligne] [Exemples]	192
10.21.7	fdf_next_field_name [Notes en ligne] [Exemples]	192
10.21.8	fdf_set_ap [Notes en ligne] [Exemples]	192
10.21.9	fdf_set_status [Notes en ligne] [Exemples]	192
10.21.10	fdf_get_status [Notes en ligne] [Exemples]	192
10.21.11	fdf_set_file [Notes en ligne] [Exemples]	193
10.21.12	fdf_get_file [Notes en ligne] [Exemples]	193

Table of Contents

10.22 Fonctions filePro [Notes en ligne]	193
10.22.1 filepro [Notes en ligne] [Exemples]	193
10.22.2 filepro fieldname [Notes en ligne] [Exemples]	193
10.22.3 filepro fieldtype [Notes en ligne] [Exemples]	193
10.22.4 filepro fieldwidth [Notes en ligne] [Exemples]	194
10.22.5 filepro retrieve [Notes en ligne] [Exemples]	194
10.22.6 filepro fieldcount [Notes en ligne] [Exemples]	194
10.22.7 filepro rowcount [Notes en ligne] [Exemples]	194
10.23 Système de fichiers [Notes en ligne]	194
10.23.1 basename [Notes en ligne] [Exemples]	194
10.23.2 chgrp [Notes en ligne] [Exemples]	195
10.23.3 chmod [Notes en ligne] [Exemples]	195
10.23.4 chown [Notes en ligne] [Exemples]	195
10.23.5 clearstatcache [Notes en ligne] [Exemples]	196
10.23.6 copy [Notes en ligne] [Exemples]	196
10.23.7 delete [Notes en ligne] [Exemples]	196
10.23.8 dirname [Notes en ligne] [Exemples]	196
10.23.9 diskfreespace [Notes en ligne] [Exemples]	197
10.23.10 fclose [Notes en ligne] [Exemples]	197
10.23.11 feof [Notes en ligne] [Exemples]	197
10.23.12 fgetc [Notes en ligne] [Exemples]	197
10.23.13 fgetcsv [Notes en ligne] [Exemples]	198
10.23.14 fgets [Notes en ligne] [Exemples]	198
10.23.15 fgetss [Notes en ligne] [Exemples]	199
10.23.16 file [Notes en ligne] [Exemples]	199
10.23.17 file_exists [Notes en ligne] [Exemples]	199
10.23.18 fileatime [Notes en ligne] [Exemples]	200
10.23.19 filectime [Notes en ligne] [Exemples]	200
10.23.20 filegroup [Notes en ligne] [Exemples]	200
10.23.21 fileinode [Notes en ligne] [Exemples]	200
10.23.22 filemtime [Notes en ligne] [Exemples]	200
10.23.23 fileowner [Notes en ligne] [Exemples]	201
10.23.24 fileperms [Notes en ligne] [Exemples]	201
10.23.25 filesize [Notes en ligne] [Exemples]	201
10.23.26 filetype [Notes en ligne] [Exemples]	201
10.23.27 flock [Notes en ligne] [Exemples]	201
10.23.28 fopen [Notes en ligne] [Exemples]	202
10.23.29 fpassthru [Notes en ligne] [Exemples]	203
10.23.30 fputs [Notes en ligne] [Exemples]	203
10.23.31 fread [Notes en ligne] [Exemples]	203
10.23.32 fscanf [Notes en ligne] [Exemples]	204
10.23.33 fseek [Notes en ligne] [Exemples]	204
10.23.34 fstat [Notes en ligne] [Exemples]	205
10.23.35 ftell [Notes en ligne] [Exemples]	205
10.23.36 ftruncate [Notes en ligne] [Exemples]	205
10.23.37 fwrite [Notes en ligne] [Exemples]	206
10.23.38 set_file_buffer [Notes en ligne] [Exemples]	206

Table of Contents

10.23.39 is_dir [Notes en ligne] [Exemples]	206
10.23.40 is_executable [Notes en ligne] [Exemples]	206
10.23.41 is_file [Notes en ligne] [Exemples]	207
10.23.42 is_link [Notes en ligne] [Exemples]	207
10.23.43 is_readable [Notes en ligne] [Exemples]	207
10.23.44 is_writable [Notes en ligne] [Exemples]	207
10.23.45 is_uploaded_file [Notes en ligne] [Exemples]	208
10.23.46 link [Notes en ligne] [Exemples]	208
10.23.47 linkinfo [Notes en ligne] [Exemples]	208
10.23.48 mkdir [Notes en ligne] [Exemples]	208
10.23.49 move_uploaded_file [Notes en ligne] [Exemples]	209
10.23.50 pclose [Notes en ligne] [Exemples]	209
10.23.51 popen [Notes en ligne] [Exemples]	209
10.23.52 readfile [Notes en ligne] [Exemples]	209
10.23.53 readlink [Notes en ligne] [Exemples]	210
10.23.54 rename [Notes en ligne] [Exemples]	210
10.23.55 rewind [Notes en ligne] [Exemples]	210
10.23.56 rmdir [Notes en ligne] [Exemples]	210
10.23.57 stat [Notes en ligne] [Exemples]	211
10.23.58 lstat [Notes en ligne] [Exemples]	211
10.23.59 realpath [Notes en ligne] [Exemples]	212
10.23.60 symlink [Notes en ligne] [Exemples]	212
10.23.61 tempnam [Notes en ligne] [Exemples]	212
10.23.62 tmpfile [Notes en ligne] [Exemples]	213
10.23.63 touch [Notes en ligne] [Exemples]	213
10.23.64 umask [Notes en ligne] [Exemples]	213
10.23.65 unlink [Notes en ligne] [Exemples]	213
10.24 FTP [Notes en ligne]	214
10.24.1 ftp_connect [Notes en ligne] [Exemples]	214
10.24.2 ftp_login [Notes en ligne] [Exemples]	214
10.24.3 ftp_pwd [Notes en ligne] [Exemples]	214
10.24.4 ftp_cdup [Notes en ligne] [Exemples]	214
10.24.5 ftp_chdir [Notes en ligne] [Exemples]	214
10.24.6 ftp_mkdir [Notes en ligne] [Exemples]	215
10.24.7 ftp_rmdir [Notes en ligne] [Exemples]	215
10.24.8 ftp_nlist [Notes en ligne] [Exemples]	215
10.24.9 ftp_rawlist [Notes en ligne] [Exemples]	215
10.24.10 ftp_systype [Notes en ligne] [Exemples]	215
10.24.11 ftp_pasv [Notes en ligne] [Exemples]	215
10.24.12 ftp_get [Notes en ligne] [Exemples]	216
10.24.13 ftp_fget [Notes en ligne] [Exemples]	216
10.24.14 ftp_put [Notes en ligne] [Exemples]	216
10.24.15 ftp_fput [Notes en ligne] [Exemples]	216
10.24.16 ftp_size [Notes en ligne] [Exemples]	216
10.24.17 ftp_mdtm [Notes en ligne] [Exemples]	217
10.24.18 ftp_rename [Notes en ligne] [Exemples]	217
10.24.19 ftp_delete [Notes en ligne] [Exemples]	217

Table of Contents

10.24.20	ftp_site [Notes en ligne] [Exemples]	217
10.24.21	ftp_quit [Notes en ligne] [Exemples]	217
10.25	Function Handling functions [Notes en ligne]	217
10.25.1	call_user_func [Notes en ligne] [Exemples]	218
10.25.2	create_function [Notes en ligne] [Exemples]	218
10.25.3	func_get_arg [Notes en ligne] [Exemples]	220
10.25.4	func_get_args [Notes en ligne] [Exemples]	220
10.25.5	func_num_args [Notes en ligne] [Exemples]	221
10.25.6	function_exists [Notes en ligne] [Exemples]	221
10.25.7	register_shutdown_function [Notes en ligne] [Exemples]	221
10.26	Fonctions GNU Gettext [Notes en ligne]	222
10.26.1	bindtextdomain [Notes en ligne] [Exemples]	222
10.26.2	dcgettext [Notes en ligne] [Exemples]	222
10.26.3	dgettext [Notes en ligne] [Exemples]	222
10.26.4	gettext [Notes en ligne] [Exemples]	222
10.26.5	textdomain [Notes en ligne] [Exemples]	223
10.27	Fonctions HTTP [Notes en ligne]	223
10.27.1	header [Notes en ligne] [Exemples]	223
10.27.2	setcookie [Notes en ligne] [Exemples]	224
10.28	Fonctions Hyperwave [Notes en ligne]	225
10.28.1	Introduction [Notes en ligne]	225
10.28.2	Intégration avec Apache [Notes en ligne]	227
10.28.3	A faire [Notes en ligne]	228
10.28.4	hw_Array2Objrec [Notes en ligne] [Exemples]	228
10.28.5	hw_Children [Notes en ligne] [Exemples]	228
10.28.6	hw_ChildrenObj [Notes en ligne] [Exemples]	229
10.28.7	hw_Close [Notes en ligne] [Exemples]	229
10.28.8	hw_Connect [Notes en ligne] [Exemples]	229
10.28.9	hw_Cp [Notes en ligne] [Exemples]	229
10.28.10	hw_Deleteobject [Notes en ligne] [Exemples]	229
10.28.11	hw_DocByAnchor [Notes en ligne] [Exemples]	230
10.28.12	hw_DocByAnchorObj [Notes en ligne] [Exemples]	230
10.28.13	hw_DocumentAttributes [Notes en ligne] [Exemples]	230
10.28.14	hw_DocumentBodyTag [Notes en ligne] [Exemples]	230
10.28.15	hw_DocumentContent [Notes en ligne] [Exemples]	230
10.28.16	hw_DocumentSetContent [Notes en ligne] [Exemples]	230
10.28.17	hw_DocumentSize [Notes en ligne] [Exemples]	231
10.28.18	hw_ErrorMsg [Notes en ligne] [Exemples]	231
10.28.19	hw_EditText [Notes en ligne] [Exemples]	231
10.28.20	hw_Error [Notes en ligne] [Exemples]	231
10.28.21	hw_Free_Document [Notes en ligne] [Exemples]	231
10.28.22	hw_GetParents [Notes en ligne] [Exemples]	232
10.28.23	hw_GetParentsObj [Notes en ligne] [Exemples]	232
10.28.24	hw_GetChildColl [Notes en ligne] [Exemples]	232
10.28.25	hw_GetChildCollObj [Notes en ligne] [Exemples]	232
10.28.26	hw_GetRemote [Notes en ligne] [Exemples]	232
10.28.27	hw_GetRemoteChildren [Notes en ligne] [Exemples]	233

Table of Contents

10.28.28 hw GetSrcByDestObj [Notes en ligne] [Exemples]	233
10.28.29 hw GetObject [Notes en ligne] [Exemples]	233
10.28.30 hw GetAndLock [Notes en ligne] [Exemples]	234
10.28.31 hw GetText [Notes en ligne] [Exemples]	234
10.28.32 hw GetObjectByQuery [Notes en ligne] [Exemples]	234
10.28.33 hw GetObjectByQueryObj [Notes en ligne] [Exemples]	235
10.28.34 hw GetObjectByQueryColl [Notes en ligne] [Exemples]	235
10.28.35 hw GetObjectByQueryCollObj [Notes en ligne] [Exemples]	235
10.28.36 hw GetChildDocColl [Notes en ligne] [Exemples]	235
10.28.37 hw GetChildDocCollObj [Notes en ligne] [Exemples]	235
10.28.38 hw GetAnchors [Notes en ligne] [Exemples]	236
10.28.39 hw GetAnchorsObj [Notes en ligne] [Exemples]	236
10.28.40 hw Mv [Notes en ligne] [Exemples]	236
10.28.41 hw Identify [Notes en ligne] [Exemples]	236
10.28.42 hw InCollections [Notes en ligne] [Exemples]	236
10.28.43 hw Info [Notes en ligne] [Exemples]	237
10.28.44 hw InsColl [Notes en ligne] [Exemples]	237
10.28.45 hw InsDoc [Notes en ligne] [Exemples]	237
10.28.46 hw InsertDocument [Notes en ligne] [Exemples]	237
10.28.47 hw InsertObject [Notes en ligne] [Exemples]	237
10.28.48 hw mapid [Notes en ligne] [Exemples]	238
10.28.49 hw Modifyobject [Notes en ligne] [Exemples]	238
10.28.50 hw New Document [Notes en ligne] [Exemples]	240
10.28.51 hw Objrec2Array [Notes en ligne] [Exemples]	240
10.28.52 hw OutputDocument [Notes en ligne] [Exemples]	240
10.28.53 hw pConnect [Notes en ligne] [Exemples]	240
10.28.54 hw PipeDocument [Notes en ligne] [Exemples]	240
10.28.55 hw Root [Notes en ligne] [Exemples]	241
10.28.56 hw Unlock [Notes en ligne] [Exemples]	241
10.28.57 hw Who [Notes en ligne] [Exemples]	241
10.28.58 hw Username [Notes en ligne] [Exemples]	241
10.29 Fonctions InterBase [Notes en ligne]	241
10.29.1 ibase connect [Notes en ligne] [Exemples]	242
10.29.2 ibase pconnect [Notes en ligne] [Exemples]	242
10.29.3 ibase close [Notes en ligne] [Exemples]	242
10.29.4 ibase query [Notes en ligne] [Exemples]	242
10.29.5 ibase fetch_row [Notes en ligne] [Exemples]	242
10.29.6 ibase fetch_object [Notes en ligne] [Exemples]	243
10.29.7 ibase free_result [Notes en ligne] [Exemples]	243
10.29.8 ibase prepare [Notes en ligne] [Exemples]	243
10.29.9 ibase bind [Notes en ligne] [Exemples]	243
10.29.10 ibase execute [Notes en ligne] [Exemples]	243
10.29.11 ibase free_query [Notes en ligne] [Exemples]	244
10.29.12 ibase timefmt [Notes en ligne] [Exemples]	244
10.29.13 ibase_num_fields [Notes en ligne] [Exemples]	244
10.30 Fonctions ICAP [Notes en ligne]	244
10.30.1 icap_open [Notes en ligne] [Exemples]	245

Table of Contents

10.30.2	icap_close [Notes en ligne] [Exemples]	245
10.30.3	icap_fetch_event [Notes en ligne] [Exemples]	245
10.30.4	icap_list_events [Notes en ligne] [Exemples]	245
10.30.5	icap_store_event [Notes en ligne] [Exemples]	246
10.30.6	icap_delete_event [Notes en ligne] [Exemples]	246
10.30.7	icap_snooze [Notes en ligne] [Exemples]	247
10.30.8	icap_list_alarms [Notes en ligne] [Exemples]	247
10.31	Fonctions Informix [Notes en ligne]	247
10.31.1	ifx_connect [Notes en ligne] [Exemples]	248
10.31.2	ifx_pconnect [Notes en ligne] [Exemples]	249
10.31.3	ifx_close [Notes en ligne] [Exemples]	249
10.31.4	ifx_query [Notes en ligne] [Exemples]	250
10.31.5	ifx_prepare [Notes en ligne] [Exemples]	251
10.31.6	ifx_do [Notes en ligne] [Exemples]	251
10.31.7	ifx_error [Notes en ligne] [Exemples]	252
10.31.8	ifx_errormsg [Notes en ligne] [Exemples]	252
10.31.9	ifx_affected_rows [Notes en ligne] [Exemples]	252
10.31.10	ifx_getsqlca [Notes en ligne] [Exemples]	253
10.31.11	ifx_fetch_row [Notes en ligne] [Exemples]	253
10.31.12	ifx_htmltbl_result [Notes en ligne] [Exemples]	254
10.31.13	ifx_fieldtypes [Notes en ligne] [Exemples]	255
10.31.14	ifx_fieldproperties [Notes en ligne] [Exemples]	255
10.31.15	ifx_num_fields [Notes en ligne] [Exemples]	255
10.31.16	ifx_num_rows [Notes en ligne] [Exemples]	256
10.31.17	ifx_free_result [Notes en ligne] [Exemples]	256
10.31.18	ifx_create_char [Notes en ligne] [Exemples]	256
10.31.19	ifx_free_char [Notes en ligne] [Exemples]	256
10.31.20	ifx_update_char [Notes en ligne] [Exemples]	256
10.31.21	ifx_get_char [Notes en ligne] [Exemples]	256
10.31.22	ifx_create_blob [Notes en ligne] [Exemples]	257
10.31.23	ifx_copy_blob [Notes en ligne] [Exemples]	257
10.31.24	ifx_free_blob [Notes en ligne] [Exemples]	257
10.31.25	ifx_get_blob [Notes en ligne] [Exemples]	257
10.31.26	ifx_update_blob [Notes en ligne] [Exemples]	257
10.31.27	ifx_blobinfile_mode [Notes en ligne] [Exemples]	257
10.31.28	ifx_textasvarchar [Notes en ligne] [Exemples]	258
10.31.29	ifx_byteasvarchar [Notes en ligne] [Exemples]	258
10.31.30	ifx_nullformat [Notes en ligne] [Exemples]	258
10.31.31	ifxus_create_slob [Notes en ligne] [Exemples]	258
10.31.32	ifx_free_slob [Notes en ligne] [Exemples]	258
10.31.33	ifxus_close_slob [Notes en ligne] [Exemples]	258
10.31.34	ifxus_open_slob [Notes en ligne] [Exemples]	259
10.31.35	ifxus_tell_slob [Notes en ligne] [Exemples]	259
10.31.36	ifxus_seek_slob [Notes en ligne] [Exemples]	259
10.31.37	ifxus_read_slob [Notes en ligne] [Exemples]	259
10.31.38	ifxus_write_slob [Notes en ligne] [Exemples]	259
10.32	Images [Notes en ligne]	260

Table of Contents

10.32.1 GetImageSize [Notes en ligne] [Exemples]	260
10.32.2 ImageArc [Notes en ligne] [Exemples]	260
10.32.3 ImageChar [Notes en ligne] [Exemples]	261
10.32.4 ImageCharUp [Notes en ligne] [Exemples]	261
10.32.5 ImageColorAllocate [Notes en ligne] [Exemples]	261
10.32.6 ImageColorDeAllocate [Notes en ligne] [Exemples]	261
10.32.7 ImageColorAt [Notes en ligne] [Exemples]	262
10.32.8 ImageColorClosest [Notes en ligne] [Exemples]	262
10.32.9 ImageColorExact [Notes en ligne] [Exemples]	262
10.32.10 ImageColorResolve [Notes en ligne] [Exemples]	262
10.32.11 ImageGammaCorrect [Notes en ligne] [Exemples]	262
10.32.12 ImageColorSet [Notes en ligne] [Exemples]	263
10.32.13 ImageColorsForIndex [Notes en ligne] [Exemples]	263
10.32.14 ImageColorsTotal [Notes en ligne] [Exemples]	263
10.32.15 ImageColorTransparent [Notes en ligne] [Exemples]	263
10.32.16 ImageCopy [Notes en ligne] [Exemples]	263
10.32.17 ImageCopyResized [Notes en ligne] [Exemples]	264
10.32.18 ImageCreate [Notes en ligne] [Exemples]	264
10.32.19 ImageCreateFromGif [Notes en ligne] [Exemples]	264
10.32.20 ImageCreateFromJPEG [Notes en ligne] [Exemples]	265
10.32.21 ImageCreateFromPNG [Notes en ligne] [Exemples]	265
10.32.22 ImageDashedLine [Notes en ligne] [Exemples]	266
10.32.23 ImageDestroy [Notes en ligne] [Exemples]	266
10.32.24 ImageFill [Notes en ligne] [Exemples]	266
10.32.25 ImageFilledPolygon [Notes en ligne] [Exemples]	266
10.32.26 ImageFilledRectangle [Notes en ligne] [Exemples]	266
10.32.27 ImageFillToBorder [Notes en ligne] [Exemples]	267
10.32.28 ImageFontHeight [Notes en ligne] [Exemples]	267
10.32.29 ImageFontWidth [Notes en ligne] [Exemples]	267
10.32.30 ImageGif [Notes en ligne] [Exemples]	267
10.32.31 ImagePNG [Notes en ligne] [Exemples]	268
10.32.32 ImageJPEG [Notes en ligne] [Exemples]	268
10.32.33 ImageInterlace [Notes en ligne] [Exemples]	269
10.32.34 ImageLine [Notes en ligne] [Exemples]	269
10.32.35 ImageLoadFont [Notes en ligne] [Exemples]	269
10.32.36 ImagePsExtendFont [Notes en ligne] [Exemples]	270
10.32.37 ImagePsSlantFont [Notes en ligne] [Exemples]	270
10.32.38 ImagePolygon [Notes en ligne] [Exemples]	270
10.32.39 ImagePSBBox [Notes en ligne] [Exemples]	270
10.32.40 ImagePSEncodeFont [Notes en ligne] [Exemples]	271
10.32.41 ImagePSFreeFont [Notes en ligne] [Exemples]	271
10.32.42 ImagePSLoadFont [Notes en ligne] [Exemples]	271
10.32.43 ImagePSText [Notes en ligne] [Exemples]	271
10.32.44 ImageRectangle [Notes en ligne] [Exemples]	272
10.32.45 ImageSetPixel [Notes en ligne] [Exemples]	272
10.32.46 ImageString [Notes en ligne] [Exemples]	272
10.32.47 ImageStringUp [Notes en ligne] [Exemples]	273

Table of Contents

10.32.48 ImageSX [Notes en ligne] [Exemples]	273
10.32.49 ImageSY [Notes en ligne] [Exemples]	273
10.32.50 ImageTTFBBox [Notes en ligne] [Exemples]	273
10.32.51 ImageTTFText [Notes en ligne] [Exemples]	274
10.32.52 ImageTypes [Notes en ligne] [Exemples]	275
10.32.53 read_exif_data [Notes en ligne] [Exemples]	275
10.33 Fonctions IMAP [Notes en ligne]	276
10.33.1 imap_append [Notes en ligne] [Exemples]	276
10.33.2 imap_base64 [Notes en ligne] [Exemples]	276
10.33.3 imap_body [Notes en ligne] [Exemples]	276
10.33.4 imap_check [Notes en ligne] [Exemples]	277
10.33.5 imap_close [Notes en ligne] [Exemples]	277
10.33.6 imap_createmailbox [Notes en ligne] [Exemples]	277
10.33.7 imap_delete [Notes en ligne] [Exemples]	278
10.33.8 imap_deletemailbox [Notes en ligne] [Exemples]	278
10.33.9 imap_expunge [Notes en ligne] [Exemples]	278
10.33.10 imap_fetchbody [Notes en ligne] [Exemples]	278
10.33.11 imap_fetchstructure [Notes en ligne] [Exemples]	279
10.33.12 imap_header [Notes en ligne] [Exemples]	280
10.33.13 imap_headers [Notes en ligne] [Exemples]	281
10.33.14 imap_listmailbox [Notes en ligne] [Exemples]	282
10.33.15 imap_getmailboxes [Notes en ligne] [Exemples]	282
10.33.16 imap_listsubscribed [Notes en ligne] [Exemples]	282
10.33.17 imap_getsubscribed [Notes en ligne] [Exemples]	282
10.33.18 imap_mail_copy [Notes en ligne] [Exemples]	283
10.33.19 imap_mail_move [Notes en ligne] [Exemples]	283
10.33.20 imap_num_msg [Notes en ligne] [Exemples]	283
10.33.21 imap_num_recent [Notes en ligne] [Exemples]	283
10.33.22 imap_open [Notes en ligne] [Exemples]	283
10.33.23 imap_ping [Notes en ligne] [Exemples]	284
10.33.24 imap_renamemailbox [Notes en ligne] [Exemples]	284
10.33.25 imap_reopen [Notes en ligne] [Exemples]	285
10.33.26 imap_subscribe [Notes en ligne] [Exemples]	285
10.33.27 imap_undelete [Notes en ligne] [Exemples]	285
10.33.28 imap_unsubscribe [Notes en ligne] [Exemples]	285
10.33.29 imap_qprint [Notes en ligne] [Exemples]	285
10.33.30 imap_8bit [Notes en ligne] [Exemples]	286
10.33.31 imap_binary [Notes en ligne] [Exemples]	286
10.33.32 imap_scanmailbox [Notes en ligne] [Exemples]	286
10.33.33 imap_mailboxmsginfo [Notes en ligne] [Exemples]	286
10.33.34 imap_rfc822_write_address [Notes en ligne] [Exemples]	286
10.33.35 imap_rfc822_parse_adrlist [Notes en ligne] [Exemples]	287
10.33.36 imap_setflag_full [Notes en ligne] [Exemples]	287
10.33.37 imap_clearflag_full [Notes en ligne] [Exemples]	287
10.33.38 imap_sort [Notes en ligne] [Exemples]	288
10.33.39 imap_fetchheader [Notes en ligne] [Exemples]	288
10.33.40 imap_uid [Notes en ligne] [Exemples]	288

Table of Contents

10.33.41	imap_msgno [Notes en ligne] [Exemples]	289
10.33.42	imap_search [Notes en ligne] [Exemples]	289
10.33.43	imap_last_error [Notes en ligne] [Exemples]	290
10.33.44	imap_errors [Notes en ligne] [Exemples]	290
10.33.45	imap_alerts [Notes en ligne] [Exemples]	290
10.33.46	imap_status [Notes en ligne] [Exemples]	290
10.33.47	imap_utf7_decode [Notes en ligne] [Exemples]	291
10.33.48	imap_utf7_encode [Notes en ligne] [Exemples]	291
10.34	options PHP & informations [Notes en ligne]	291
10.34.1	extension_loaded [Notes en ligne] [Exemples]	291
10.34.2	getenv [Notes en ligne] [Exemples]	291
10.34.3	get_cfg_var [Notes en ligne] [Exemples]	291
10.34.4	get_current_user [Notes en ligne] [Exemples]	292
10.34.5	get_magic_quotes_gpc [Notes en ligne] [Exemples]	292
10.34.6	get_magic_quotes_runtime [Notes en ligne] [Exemples]	292
10.34.7	getlastmod [Notes en ligne] [Exemples]	292
10.34.8	getmyinode [Notes en ligne] [Exemples]	293
10.34.9	getmypid [Notes en ligne] [Exemples]	293
10.34.10	getmyuid [Notes en ligne] [Exemples]	293
10.34.11	getrusage [Notes en ligne] [Exemples]	293
10.34.12	phpinfo [Notes en ligne] [Exemples]	294
10.34.13	phpversion [Notes en ligne] [Exemples]	294
10.34.14	php_logo_guid [Notes en ligne] [Exemples]	294
10.34.15	putenv [Notes en ligne] [Exemples]	294
10.34.16	set_magic_quotes_runtime [Notes en ligne] [Exemples]	295
10.34.17	set_time_limit [Notes en ligne] [Exemples]	295
10.34.18	zend_logo_guid [Notes en ligne] [Exemples]	295
10.35	Fonctions LDAP [Notes en ligne]	295
10.35.1	Introduction à LDAP [Notes en ligne]	295
10.35.2	Exemple complet [Notes en ligne]	296
10.35.2.1	Utilisation des fonctions PHP LDAP [Notes en ligne]	297
10.35.2.2	Plus d'informations [Notes en ligne]	297
10.35.3	ldap_add [Notes en ligne] [Exemples]	297
10.35.4	ldap_mod_add [Notes en ligne] [Exemples]	298
10.35.5	ldap_mod_del [Notes en ligne] [Exemples]	298
10.35.6	ldap_mod_replace [Notes en ligne] [Exemples]	298
10.35.7	ldap_bind [Notes en ligne] [Exemples]	299
10.35.8	ldap_close [Notes en ligne] [Exemples]	299
10.35.9	ldap_connect [Notes en ligne] [Exemples]	299
10.35.10	ldap_count_entries [Notes en ligne] [Exemples]	299
10.35.11	ldap_delete [Notes en ligne] [Exemples]	300
10.35.12	ldap_dn2ufn [Notes en ligne] [Exemples]	300
10.35.13	ldap_explode_dn [Notes en ligne] [Exemples]	300
10.35.14	ldap_first_attribute [Notes en ligne] [Exemples]	300
10.35.15	ldap_first_entry [Notes en ligne] [Exemples]	300
10.35.16	ldap_free_result [Notes en ligne] [Exemples]	301
10.35.17	ldap_get_attributes [Notes en ligne] [Exemples]	301

Table of Contents

10.35.18 ldap_get_dn [Notes en ligne] [Exemples]	301
10.35.19 ldap_get_entries [Notes en ligne] [Exemples]	302
10.35.20 ldap_get_values [Notes en ligne] [Exemples]	302
10.35.21 ldap_get_values_len [Notes en ligne] [Exemples]	303
10.35.22 ldap_list [Notes en ligne] [Exemples]	303
10.35.23 ldap_modify [Notes en ligne] [Exemples]	303
10.35.24 ldap_next_attribute [Notes en ligne] [Exemples]	304
10.35.25 ldap_next_entry [Notes en ligne] [Exemples]	304
10.35.26 ldap_read [Notes en ligne] [Exemples]	304
10.35.27 ldap_search [Notes en ligne] [Exemples]	304
10.35.28 ldap_unbind [Notes en ligne] [Exemples]	305
10.35.29 ldap_err2str [Notes en ligne] [Exemples]	305
10.35.30 ldap_errno [Notes en ligne] [Exemples]	306
10.35.31 ldap_error [Notes en ligne] [Exemples]	306
10.36 Fonction mail [Notes en ligne]	307
10.36.1 mail [Notes en ligne] [Exemples]	307
10.36.2 ezmlm_hash [Notes en ligne] [Exemples]	308
10.37 Fonctions mathématiques [Notes en ligne]	308
10.37.1 Introduction [Notes en ligne]	308
10.37.1.1 Constantes mathématiques [Notes en ligne]	308
10.37.2 Abs [Notes en ligne] [Exemples]	308
10.37.3 Acos [Notes en ligne] [Exemples]	309
10.37.4 Asin [Notes en ligne] [Exemples]	309
10.37.5 Atan [Notes en ligne] [Exemples]	309
10.37.6 Atan2 [Notes en ligne] [Exemples]	309
10.37.7 base_convert [Notes en ligne] [Exemples]	309
10.37.8 BinDec [Notes en ligne] [Exemples]	310
10.37.9 Ceil [Notes en ligne] [Exemples]	310
10.37.10 Cos [Notes en ligne] [Exemples]	310
10.37.11 DecBin [Notes en ligne] [Exemples]	310
10.37.12 DecHex [Notes en ligne] [Exemples]	311
10.37.13 DecOct [Notes en ligne] [Exemples]	311
10.37.14 Exp [Notes en ligne] [Exemples]	311
10.37.15 Floor [Notes en ligne] [Exemples]	311
10.37.16 getrandmax [Notes en ligne] [Exemples]	311
10.37.17 HexDec [Notes en ligne] [Exemples]	312
10.37.18 Log [Notes en ligne] [Exemples]	312
10.37.19 Log10 [Notes en ligne] [Exemples]	312
10.37.20 max [Notes en ligne] [Exemples]	312
10.37.21 min [Notes en ligne] [Exemples]	312
10.37.22 mt_rand [Notes en ligne] [Exemples]	313
10.37.23 mt_srand [Notes en ligne] [Exemples]	313
10.37.24 mt_getrandmax [Notes en ligne] [Exemples]	313
10.37.25 number_format [Notes en ligne] [Exemples]	313
10.37.26 OctDec [Notes en ligne] [Exemples]	314
10.37.27 pi [Notes en ligne] [Exemples]	314
10.37.28 pow [Notes en ligne] [Exemples]	314

Table of Contents

10.37.29 rand [Notes en ligne] [Exemples]	314
10.37.30 round [Notes en ligne] [Exemples]	315
10.37.31 Sin [Notes en ligne] [Exemples]	315
10.37.32 Sqrt [Notes en ligne] [Exemples]	315
10.37.33 srand [Notes en ligne] [Exemples]	315
10.37.34 Tan [Notes en ligne] [Exemples]	315
10.38 Fonctions MCAL [Notes en ligne]	316
10.38.1 mcal_open [Notes en ligne] [Exemples]	316
10.38.2 mcal_close [Notes en ligne] [Exemples]	316
10.38.3 mcal_fetch_event [Notes en ligne] [Exemples]	317
10.38.4 mcal_list_events [Notes en ligne] [Exemples]	317
10.38.5 mcal_append_event [Notes en ligne] [Exemples]	317
10.38.6 mcal_store_event [Notes en ligne] [Exemples]	318
10.38.7 mcal_delete_event [Notes en ligne] [Exemples]	318
10.38.8 mcal_snooze [Notes en ligne] [Exemples]	318
10.38.9 mcal_list_alarms [Notes en ligne] [Exemples]	318
10.38.10 mcal_event_init [Notes en ligne] [Exemples]	318
10.38.11 mcal_event_set_category [Notes en ligne] [Exemples]	319
10.38.12 mcal_event_set_title [Notes en ligne] [Exemples]	319
10.38.13 mcal_event_set_description [Notes en ligne] [Exemples]	319
10.38.14 mcal_event_set_start [Notes en ligne] [Exemples]	319
10.38.15 mcal_event_set_end [Notes en ligne] [Exemples]	319
10.38.16 mcal_event_set_alarm [Notes en ligne] [Exemples]	319
10.38.17 mcal_event_set_class [Notes en ligne] [Exemples]	320
10.38.18 mcal_is_leap_year [Notes en ligne] [Exemples]	320
10.38.19 mcal_days_in_month [Notes en ligne] [Exemples]	320
10.38.20 mcal_date_valid [Notes en ligne] [Exemples]	320
10.38.21 mcal_time_valid [Notes en ligne] [Exemples]	320
10.38.22 mcal_day_of_week [Notes en ligne] [Exemples]	321
10.38.23 mcal_day_of_year [Notes en ligne] [Exemples]	321
10.38.24 mcal_date_compare [Notes en ligne] [Exemples]	321
10.38.25 mcal_next_recurrence [Notes en ligne] [Exemples]	321
10.38.26 mcal_event_set_recur_none [Notes en ligne] [Exemples]	321
10.38.27 mcal_event_set_recur_daily [Notes en ligne] [Exemples]	321
10.38.28 mcal_event_set_recur_weekly [Notes en ligne] [Exemples]	322
10.38.29 mcal_event_set_recur_monthly_mday [Notes en ligne] [Exemples]	322
10.38.30 mcal_event_set_recur_monthly_wday [Notes en ligne] [Exemples]	322
10.38.31 mcal_event_set_recur_yearly [Notes en ligne] [Exemples]	322
10.38.32 mcal_fetch_current_stream_event [Notes en ligne] [Exemples]	322
10.39 Cryptage [Notes en ligne]	323
10.39.1 mcrypt_get_cipher_name [Notes en ligne] [Exemples]	325
10.39.2 mcrypt_get_block_size [Notes en ligne] [Exemples]	326
10.39.3 mcrypt_get_key_size [Notes en ligne] [Exemples]	326
10.39.4 mcrypt_create_iv [Notes en ligne] [Exemples]	326
10.39.5 mcrypt_cbc [Notes en ligne] [Exemples]	327
10.39.6 mcrypt_cfb [Notes en ligne] [Exemples]	327
10.39.7 mcrypt_ecb [Notes en ligne] [Exemples]	327

Table of Contents

10.39.8 mdecrypt_ofb [Notes en ligne] [Exemples].....	328
10.39.9 mdecrypt_list_algorithms [Notes en ligne] [Exemples].....	328
10.39.10 mdecrypt_list_modes [Notes en ligne] [Exemples].....	329
10.39.11 mdecrypt_get_iv_size [Notes en ligne] [Exemples].....	329
10.39.12 mdecrypt_encrypt [Notes en ligne] [Exemples].....	329
10.39.13 mdecrypt_decrypt [Notes en ligne] [Exemples].....	330
10.39.14 mdecrypt_module_open [Notes en ligne] [Exemples].....	330
10.39.15 mdecrypt_generic_init [Notes en ligne] [Exemples].....	331
10.39.16 mdecrypt_generic [Notes en ligne] [Exemples].....	331
10.39.17 mdecrypt_generic [Notes en ligne] [Exemples].....	331
10.39.18 mdecrypt_generic_end [Notes en ligne] [Exemples].....	332
10.39.19 mdecrypt_enc_self_test [Notes en ligne] [Exemples].....	332
10.39.20 mdecrypt_enc_is_block_algorithm_mode [Notes en ligne] [Exemples].....	332
10.39.21 mdecrypt_enc_is_block_algorithm [Notes en ligne] [Exemples].....	332
10.39.22 mdecrypt_enc_is_block_mode [Notes en ligne] [Exemples].....	333
10.39.23 mdecrypt_enc_get_block_size [Notes en ligne] [Exemples].....	333
10.39.24 mdecrypt_enc_get_key_size [Notes en ligne] [Exemples].....	333
10.39.25 mdecrypt_enc_get_supported_key_sizes [Notes en ligne] [Exemples].....	333
10.39.26 mdecrypt_enc_get_iv_size [Notes en ligne] [Exemples].....	333
10.39.27 mdecrypt_enc_get_algorithms_name [Notes en ligne] [Exemples].....	333
10.39.28 mdecrypt_enc_get_modes_name [Notes en ligne] [Exemples].....	334
10.39.29 mdecrypt_module_self_test [Notes en ligne] [Exemples].....	334
10.39.30 mdecrypt_module_is_block_algorithm_mode [Notes en ligne] [Exemples].....	334
10.39.31 mdecrypt_module_is_block_algorithm [Notes en ligne] [Exemples].....	334
10.39.32 mdecrypt_module_is_block_mode [Notes en ligne] [Exemples].....	334
10.39.33 mdecrypt_module_get_algo_block_size [Notes en ligne] [Exemples].....	334
10.39.34 mdecrypt_module_get_algo_key_size [Notes en ligne] [Exemples].....	335
10.39.35 mdecrypt_module_get_algo_supported_key_sizes [Notes en ligne] [Exemples].....	335
10.40 Hash [Notes en ligne]	335
10.40.1 mhash_get_hash_name [Notes en ligne] [Exemples].....	336
10.40.2 mhash_get_block_size [Notes en ligne] [Exemples].....	336
10.40.3 mhash_count [Notes en ligne] [Exemples].....	337
10.40.4 mhash [Notes en ligne] [Exemples].....	337
10.40.5 mhash_keygen_s2k [Notes en ligne] [Exemples].....	337
10.41 Fonctions diverses [Notes en ligne]	337
10.41.1 connection_aborted [Notes en ligne] [Exemples].....	338
10.41.2 connection_status [Notes en ligne] [Exemples].....	338
10.41.3 connection_timeout [Notes en ligne] [Exemples].....	338
10.41.4 define [Notes en ligne] [Exemples].....	338
10.41.5 defined [Notes en ligne] [Exemples].....	339
10.41.6 die [Notes en ligne] [Exemples].....	339
10.41.7 eval [Notes en ligne] [Exemples].....	339
10.41.8 exit [Notes en ligne] [Exemples].....	340
10.41.9 get_browser [Notes en ligne] [Exemples].....	340
10.41.10 ignore_user_abort [Notes en ligne] [Exemples].....	341
10.41.11 iptcp_parse [Notes en ligne] [Exemples].....	341
10.41.12 leak [Notes en ligne] [Exemples].....	341

Table of Contents

10.41.13 pack [Notes en ligne] [Exemples]	342
10.41.14 serialize [Notes en ligne] [Exemples]	343
10.41.15 sleep [Notes en ligne] [Exemples]	343
10.41.16 uniqid [Notes en ligne] [Exemples]	343
10.41.17 unpack [Notes en ligne] [Exemples]	344
10.41.18 unserialize [Notes en ligne] [Exemples]	344
10.41.19 usleep [Notes en ligne] [Exemples]	345
10.42 fonctions mSQL [Notes en ligne]	345
10.42.1 msql [Notes en ligne] [Exemples]	345
10.42.2 msql affected_rows [Notes en ligne] [Exemples]	345
10.42.3 msql_close [Notes en ligne] [Exemples]	346
10.42.4 msql_connect [Notes en ligne] [Exemples]	346
10.42.5 msql_create_db [Notes en ligne] [Exemples]	346
10.42.6 msql_createdb [Notes en ligne] [Exemples]	346
10.42.7 msql_data_seek [Notes en ligne] [Exemples]	346
10.42.8 msql_dbname [Notes en ligne] [Exemples]	347
10.42.9 msql_drop_db [Notes en ligne] [Exemples]	347
10.42.10 msql_dropdb [Notes en ligne] [Exemples]	347
10.42.11 msql_error [Notes en ligne] [Exemples]	347
10.42.12 msql_fetch_array [Notes en ligne] [Exemples]	347
10.42.13 msql_fetch_field [Notes en ligne] [Exemples]	348
10.42.14 msql_fetch_object [Notes en ligne] [Exemples]	348
10.42.15 msql_fetch_row [Notes en ligne] [Exemples]	348
10.42.16 msql_fieldname [Notes en ligne] [Exemples]	349
10.42.17 msql_field_seek [Notes en ligne] [Exemples]	349
10.42.18 msql_fieldtable [Notes en ligne] [Exemples]	349
10.42.19 msql_fieldtype [Notes en ligne] [Exemples]	349
10.42.20 msql_fieldflags [Notes en ligne] [Exemples]	349
10.42.21 msql_fieldlen [Notes en ligne] [Exemples]	350
10.42.22 msql_free_result [Notes en ligne] [Exemples]	350
10.42.23 msql_freeresult [Notes en ligne] [Exemples]	350
10.42.24 msql_list_fields [Notes en ligne] [Exemples]	350
10.42.25 msql_listfields [Notes en ligne] [Exemples]	350
10.42.26 msql_list_dbs [Notes en ligne] [Exemples]	350
10.42.27 msql_listdbs [Notes en ligne] [Exemples]	351
10.42.28 msql_list_tables [Notes en ligne] [Exemples]	351
10.42.29 msql_listtables [Notes en ligne] [Exemples]	351
10.42.30 msql_num_fields [Notes en ligne] [Exemples]	351
10.42.31 msql_num_rows [Notes en ligne] [Exemples]	351
10.42.32 msql_numfields [Notes en ligne] [Exemples]	351
10.42.33 msql_numrows [Notes en ligne] [Exemples]	351
10.42.34 msql_pconnect [Notes en ligne] [Exemples]	352
10.42.35 msql_query [Notes en ligne] [Exemples]	352
10.42.36 msql_regcase [Notes en ligne] [Exemples]	352
10.42.37 msql_result [Notes en ligne] [Exemples]	352
10.42.38 msql_select_db [Notes en ligne] [Exemples]	353
10.42.39 msql_selectdb [Notes en ligne] [Exemples]	353

Table of Contents

10.42.40	mysql_tablename [Notes en ligne] [Exemples]	353
10.43	Fonctions Microsoft SQL Server [Notes en ligne]	353
10.43.1	mssql_close [Notes en ligne] [Exemples]	353
10.43.2	mssql_connect [Notes en ligne] [Exemples]	354
10.43.3	mssql_data_seek [Notes en ligne] [Exemples]	354
10.43.4	mssql_fetch_array [Notes en ligne] [Exemples]	354
10.43.5	mssql_fetch_field [Notes en ligne] [Exemples]	355
10.43.6	mssql_fetch_object [Notes en ligne] [Exemples]	355
10.43.7	mssql_fetch_row [Notes en ligne] [Exemples]	355
10.43.8	mssql_field_length [Notes en ligne] [Exemples]	355
10.43.9	mssql_field_name [Notes en ligne] [Exemples]	356
10.43.10	mssql_field_seek [Notes en ligne] [Exemples]	356
10.43.11	mssql_field_type [Notes en ligne] [Exemples]	356
10.43.12	mssql_free_result [Notes en ligne] [Exemples]	356
10.43.13	mssql_get_last_message [Notes en ligne] [Exemples]	356
10.43.14	mssql_min_error_severity [Notes en ligne] [Exemples]	356
10.43.15	mssql_min_message_severity [Notes en ligne] [Exemples]	356
10.43.16	mssql_num_fields [Notes en ligne] [Exemples]	357
10.43.17	mssql_num_rows [Notes en ligne] [Exemples]	357
10.43.18	mssql_pconnect [Notes en ligne] [Exemples]	357
10.43.19	mssql_query [Notes en ligne] [Exemples]	357
10.43.20	mssql_result [Notes en ligne] [Exemples]	357
10.43.21	mssql_select_db [Notes en ligne] [Exemples]	358
10.44	Fonctions MySQL [Notes en ligne]	358
10.44.1	mysql_affected_rows [Notes en ligne] [Exemples]	358
10.44.2	mysql_change_user [Notes en ligne] [Exemples]	358
10.44.3	mysql_close [Notes en ligne] [Exemples]	359
10.44.4	mysql_connect [Notes en ligne] [Exemples]	359
10.44.5	mysql_create_db [Notes en ligne] [Exemples]	360
10.44.6	mysql_data_seek [Notes en ligne] [Exemples]	360
10.44.7	mysql_db_query [Notes en ligne] [Exemples]	361
10.44.8	mysql_drop_db [Notes en ligne] [Exemples]	361
10.44.9	mysql_errno [Notes en ligne] [Exemples]	361
10.44.10	mysql_error [Notes en ligne] [Exemples]	362
10.44.11	mysql_fetch_array [Notes en ligne] [Exemples]	362
10.44.12	mysql_fetch_field [Notes en ligne] [Exemples]	363
10.44.13	mysql_fetch_lengths [Notes en ligne] [Exemples]	363
10.44.14	mysql_fetch_object [Notes en ligne] [Exemples]	364
10.44.15	mysql_fetch_row [Notes en ligne] [Exemples]	364
10.44.16	mysql_field_name [Notes en ligne] [Exemples]	365
10.44.17	mysql_field_seek [Notes en ligne] [Exemples]	365
10.44.18	mysql_field_table [Notes en ligne] [Exemples]	365
10.44.19	mysql_field_type [Notes en ligne] [Exemples]	365
10.44.20	mysql_field_flags [Notes en ligne] [Exemples]	366
10.44.21	mysql_field_len [Notes en ligne] [Exemples]	366
10.44.22	mysql_free_result [Notes en ligne] [Exemples]	366
10.44.23	mysql_insert_id [Notes en ligne] [Exemples]	366

Table of Contents

10.44.24	mysql_list_fields [Notes en ligne] [Exemples]	367
10.44.25	mysql_list_dbs [Notes en ligne] [Exemples]	367
10.44.26	mysql_list_tables [Notes en ligne] [Exemples]	367
10.44.27	mysql_num_fields [Notes en ligne] [Exemples]	367
10.44.28	mysql_num_rows [Notes en ligne] [Exemples]	368
10.44.29	mysql_pconnect [Notes en ligne] [Exemples]	368
10.44.30	mysql_query [Notes en ligne] [Exemples]	368
10.44.31	mysql_result [Notes en ligne] [Exemples]	369
10.44.32	mysql_select_db [Notes en ligne] [Exemples]	369
10.44.33	mysql_tablename [Notes en ligne] [Exemples]	370
10.45	Réseau [Notes en ligne]	370
10.45.1	checkdnsrr [Notes en ligne] [Exemples]	370
10.45.2	closelog [Notes en ligne] [Exemples]	370
10.45.3	debugger_off [Notes en ligne] [Exemples]	371
10.45.4	debugger_on [Notes en ligne] [Exemples]	371
10.45.5	define_syslog_variables [Notes en ligne] [Exemples]	371
10.45.6	fsockopen [Notes en ligne] [Exemples]	371
10.45.7	gethostbyaddr [Notes en ligne] [Exemples]	372
10.45.8	gethostbyname [Notes en ligne] [Exemples]	372
10.45.9	gethostbyname_l [Notes en ligne] [Exemples]	372
10.45.10	getmxrr [Notes en ligne] [Exemples]	372
10.45.11	getprotobyname [Notes en ligne] [Exemples]	373
10.45.12	getprotobyname [Notes en ligne] [Exemples]	373
10.45.13	getservbyname [Notes en ligne] [Exemples]	373
10.45.14	getservbyport [Notes en ligne] [Exemples]	373
10.45.15	ip2long [Notes en ligne] [Exemples]	373
10.45.16	long2ip [Notes en ligne] [Exemples]	374
10.45.17	openlog [Notes en ligne] [Exemples]	374
10.45.18	pfsockopen [Notes en ligne] [Exemples]	374
10.45.19	socket_get_status [Notes en ligne] [Exemples]	374
10.45.20	set_socket_blocking [Notes en ligne] [Exemples]	375
10.45.21	socket_set_timeout [Notes en ligne] [Exemples]	375
10.45.22	syslog [Notes en ligne] [Exemples]	375
10.46	Fonctions NIS [Notes en ligne]	376
10.46.1	yp_get_default_domain [Notes en ligne] [Exemples]	376
10.46.2	yp_order [Notes en ligne] [Exemples]	376
10.46.3	yp_master [Notes en ligne] [Exemples]	377
10.46.4	yp_match [Notes en ligne] [Exemples]	377
10.46.5	yp_first [Notes en ligne] [Exemples]	377
10.46.6	yp_next [Notes en ligne] [Exemples]	378
10.47	Oracle 8 functions [Notes en ligne]	378
10.47.1	OCIDefineByName [Notes en ligne] [Exemples]	378
10.47.2	OCIBindByName [Notes en ligne] [Exemples]	379
10.47.3	OCILogon [Notes en ligne] [Exemples]	380
10.47.4	OCIPLogon [Notes en ligne] [Exemples]	381
10.47.5	OCINLogon [Notes en ligne] [Exemples]	381
10.47.6	OCILogOff [Notes en ligne] [Exemples]	383

Table of Contents

10.47.7 OCIExecute [Notes en ligne] [Exemples]	383
10.47.8 OCICommit [Notes en ligne] [Exemples]	383
10.47.9 OCIRollback [Notes en ligne] [Exemples]	383
10.47.10 OCINewDescriptor [Notes en ligne] [Exemples]	384
10.47.11 OCIRowCount [Notes en ligne] [Exemples]	385
10.47.12 OCINumCols [Notes en ligne] [Exemples]	385
10.47.13 OCIResult [Notes en ligne] [Exemples]	386
10.47.14 OCIFetch [Notes en ligne] [Exemples]	386
10.47.15 OCIFetchInto [Notes en ligne] [Exemples]	386
10.47.16 OCIFetchStatement [Notes en ligne] [Exemples]	387
10.47.17 OCIColumnIsNULL [Notes en ligne] [Exemples]	387
10.47.18 OCIColumnSize [Notes en ligne] [Exemples]	388
10.47.19 OCIServerVersion [Notes en ligne] [Exemples]	388
10.47.20 OCIStatementType [Notes en ligne] [Exemples]	389
10.47.21 OCINewCursor [Notes en ligne] [Exemples]	389
10.47.22 OCIFreeStatement [Notes en ligne] [Exemples]	390
10.47.23 OCIFreeCursor [Notes en ligne] [Exemples]	391
10.47.24 OCIColumnName [Notes en ligne] [Exemples]	391
10.47.25 OCIColumnType [Notes en ligne] [Exemples]	392
10.47.26 OCIParse [Notes en ligne] [Exemples]	392
10.47.27 OCIError [Notes en ligne] [Exemples]	392
10.47.28 OCIInternalDebug [Notes en ligne] [Exemples]	393
10.48 Fonctions Oracle [Notes en ligne]	393
10.48.1 Ora Bind [Notes en ligne] [Exemples]	393
10.48.2 Ora Close [Notes en ligne] [Exemples]	393
10.48.3 Ora ColumnName [Notes en ligne] [Exemples]	394
10.48.4 Ora ColumnType [Notes en ligne] [Exemples]	394
10.48.5 Ora Commit [Notes en ligne] [Exemples]	394
10.48.6 Ora CommitOff [Notes en ligne] [Exemples]	394
10.48.7 Ora CommitOn [Notes en ligne] [Exemples]	395
10.48.8 Ora Error [Notes en ligne] [Exemples]	395
10.48.9 Ora ErrorCode [Notes en ligne] [Exemples]	395
10.48.10 Ora Exec [Notes en ligne] [Exemples]	395
10.48.11 Ora Fetch [Notes en ligne] [Exemples]	396
10.48.12 Ora GetColumn [Notes en ligne] [Exemples]	396
10.48.13 Ora Logoff [Notes en ligne] [Exemples]	396
10.48.14 Ora Logon [Notes en ligne] [Exemples]	396
10.48.15 Ora Open [Notes en ligne] [Exemples]	397
10.48.16 Ora Parse [Notes en ligne] [Exemples]	397
10.48.17 Ora Rollback [Notes en ligne] [Exemples]	397
10.49 Output Control Functions [Notes en ligne]	397
10.49.1 flush [Notes en ligne] [Exemples]	398
10.49.2 ob_start [Notes en ligne] [Exemples]	398
10.49.3 ob_get_contents [Notes en ligne] [Exemples]	398
10.49.4 ob_get_length [Notes en ligne] [Exemples]	398
10.49.5 ob_end_flush [Notes en ligne] [Exemples]	398
10.49.6 ob_end_clean [Notes en ligne] [Exemples]	399

Table of Contents

10.49.7	ob_implicit_flush [Notes en ligne] [Exemples].....	399
10.50	Ovrimos SQL [Notes en ligne].....	399
10.50.1	ovrimos_connect [Notes en ligne] [Exemples].....	400
10.50.2	ovrimos_close [Notes en ligne] [Exemples].....	400
10.50.3	ovrimos_close_all [Notes en ligne] [Exemples].....	400
10.50.4	ovrimos_longreadlen [Notes en ligne] [Exemples].....	400
10.50.5	ovrimos_prepare [Notes en ligne] [Exemples].....	401
10.50.6	ovrimos_execute [Notes en ligne] [Exemples].....	401
10.50.7	ovrimos_cursor [Notes en ligne] [Exemples].....	402
10.50.8	ovrimos_exec [Notes en ligne] [Exemples].....	402
10.50.9	ovrimos_fetch_into [Notes en ligne] [Exemples].....	402
10.50.10	ovrimos_fetch_row [Notes en ligne] [Exemples].....	403
10.50.11	ovrimos_result [Notes en ligne] [Exemples].....	403
10.50.12	ovrimos_result_all [Notes en ligne] [Exemples].....	404
10.50.13	ovrimos_num_rows [Notes en ligne] [Exemples].....	405
10.50.14	ovrimos_num_fields [Notes en ligne] [Exemples].....	405
10.50.15	ovrimos_field_name [Notes en ligne] [Exemples].....	405
10.50.16	ovrimos_field_type [Notes en ligne] [Exemples].....	406
10.50.17	ovrimos_field_len [Notes en ligne] [Exemples].....	406
10.50.18	ovrimos_field_num [Notes en ligne] [Exemples].....	406
10.50.19	ovrimos_free_result [Notes en ligne] [Exemples].....	406
10.50.20	ovrimos_commit [Notes en ligne] [Exemples].....	406
10.50.21	ovrimos_rollback [Notes en ligne] [Exemples].....	406
10.51	Expressions régulières compatibles Perl [Notes en ligne].....	407
10.51.1	preg_match [Notes en ligne] [Exemples].....	407
10.51.2	preg_match_all [Notes en ligne] [Exemples].....	408
10.51.3	preg_replace [Notes en ligne] [Exemples].....	409
10.51.4	preg_split [Notes en ligne] [Exemples].....	410
10.51.5	preg_quote [Notes en ligne] [Exemples].....	410
10.51.6	preg_grep [Notes en ligne] [Exemples].....	410
10.51.7	Options de recherche [Notes en ligne].....	411
10.51.8	Syntaxe des masques [Notes en ligne].....	412
10.52	Fonctions PDF [Notes en ligne].....	423
10.52.1	PDF_get_info [Notes en ligne] [Exemples].....	427
10.52.2	PDF_set_info_creator [Notes en ligne] [Exemples].....	427
10.52.3	PDF_set_info_title [Notes en ligne] [Exemples].....	427
10.52.4	PDF_set_info_subject [Notes en ligne] [Exemples].....	428
10.52.5	PDF_set_info_keywords [Notes en ligne] [Exemples].....	428
10.52.6	PDF_set_info_author [Notes en ligne] [Exemples].....	428
10.52.7	PDF_open [Notes en ligne] [Exemples].....	429
10.52.8	PDF_close [Notes en ligne] [Exemples].....	429
10.52.9	PDF_begin_page [Notes en ligne] [Exemples].....	429
10.52.10	PDF_end_page [Notes en ligne] [Exemples].....	429
10.52.11	PDF_show [Notes en ligne] [Exemples].....	430
10.52.12	PDF_show_boxed [Notes en ligne] [Exemples].....	430
10.52.13	PDF_show_xy [Notes en ligne] [Exemples].....	430
10.52.14	PDF_set_font [Notes en ligne] [Exemples].....	430

Table of Contents

10.52.15 PDF set_leading [Notes en ligne] [Exemples]	431
10.52.16 PDF set_parameter [Notes en ligne] [Exemples]	431
10.52.17 PDF set_text_rendering [Notes en ligne] [Exemples]	431
10.52.18 PDF set_horiz_scaling [Notes en ligne] [Exemples]	431
10.52.19 PDF set_text_rise [Notes en ligne] [Exemples]	431
10.52.20 PDF set_text_matrix [Notes en ligne] [Exemples]	431
10.52.21 PDF set_text_pos [Notes en ligne] [Exemples]	432
10.52.22 PDF set_char_spacing [Notes en ligne] [Exemples]	432
10.52.23 PDF set_word_spacing [Notes en ligne] [Exemples]	432
10.52.24 PDF skew [Notes en ligne] [Exemples]	432
10.52.25 PDF continue_text [Notes en ligne] [Exemples]	432
10.52.26 PDF stringwidth [Notes en ligne] [Exemples]	433
10.52.27 PDF save [Notes en ligne] [Exemples]	433
10.52.28 PDF restore [Notes en ligne] [Exemples]	433
10.52.29 PDF translate [Notes en ligne] [Exemples]	433
10.52.30 PDF scale [Notes en ligne] [Exemples]	434
10.52.31 PDF rotate [Notes en ligne] [Exemples]	434
10.52.32 PDF setflat [Notes en ligne] [Exemples]	434
10.52.33 PDF setlinejoin [Notes en ligne] [Exemples]	434
10.52.34 PDF setlinecap [Notes en ligne] [Exemples]	435
10.52.35 PDF setmiterlimit [Notes en ligne] [Exemples]	435
10.52.36 PDF setlinewidth [Notes en ligne] [Exemples]	435
10.52.37 PDF setdash [Notes en ligne] [Exemples]	435
10.52.38 PDF moveto [Notes en ligne] [Exemples]	435
10.52.39 PDF curveto [Notes en ligne] [Exemples]	435
10.52.40 PDF lineto [Notes en ligne] [Exemples]	436
10.52.41 PDF circle [Notes en ligne] [Exemples]	436
10.52.42 PDF arc [Notes en ligne] [Exemples]	436
10.52.43 PDF rect [Notes en ligne] [Exemples]	436
10.52.44 PDF closepath [Notes en ligne] [Exemples]	436
10.52.45 PDF stroke [Notes en ligne] [Exemples]	437
10.52.46 PDF closepath_stroke [Notes en ligne] [Exemples]	437
10.52.47 PDF fill [Notes en ligne] [Exemples]	437
10.52.48 PDF fill_stroke [Notes en ligne] [Exemples]	437
10.52.49 PDF closepath_fill_stroke [Notes en ligne] [Exemples]	437
10.52.50 PDF endpath [Notes en ligne] [Exemples]	438
10.52.51 PDF clip [Notes en ligne] [Exemples]	438
10.52.52 PDF setgray_fill [Notes en ligne] [Exemples]	438
10.52.53 PDF setgray_stroke [Notes en ligne] [Exemples]	438
10.52.54 PDF setgray [Notes en ligne] [Exemples]	438
10.52.55 PDF setrgbcolor_fill [Notes en ligne] [Exemples]	438
10.52.56 PDF setrgbcolor_stroke [Notes en ligne] [Exemples]	439
10.52.57 PDF setrgbcolor [Notes en ligne] [Exemples]	439
10.52.58 PDF add_outline [Notes en ligne] [Exemples]	439
10.52.59 PDF set_transition [Notes en ligne] [Exemples]	439
10.52.60 PDF set_duration [Notes en ligne] [Exemples]	440
10.52.61 PDF open_gif [Notes en ligne] [Exemples]	440

Table of Contents

10.52.62 PDF open_memory_image [Notes en ligne] [Exemples]	440
10.52.63 PDF open_jpeg [Notes en ligne] [Exemples]	441
10.52.64 PDF close_image [Notes en ligne] [Exemples]	441
10.52.65 PDF place_image [Notes en ligne] [Exemples]	441
10.52.66 PDF put_image [Notes en ligne] [Exemples]	441
10.52.67 PDF execute_image [Notes en ligne] [Exemples]	442
10.52.68 pdf add_annotation [Notes en ligne] [Exemples]	442
10.53 Verisign Payflow Pro functions [Notes en ligne]	442
10.53.1 pfpro_init [Notes en ligne] [Exemples]	443
10.53.2 pfpro_cleanup [Notes en ligne] [Exemples]	443
10.53.3 pfpro_process [Notes en ligne] [Exemples]	443
10.53.4 pfpro_process_raw [Notes en ligne] [Exemples]	444
10.53.5 pfpro_version [Notes en ligne] [Exemples]	444
10.54 Fonctions PostgreSQL [Notes en ligne]	445
10.54.1 pg_Close [Notes en ligne] [Exemples]	446
10.54.2 pg_cmdTuples [Notes en ligne] [Exemples]	446
10.54.3 pg_Connect [Notes en ligne] [Exemples]	446
10.54.4 pg_DBname [Notes en ligne] [Exemples]	447
10.54.5 pg_ErrorMessage [Notes en ligne] [Exemples]	447
10.54.6 pg_Exec [Notes en ligne] [Exemples]	447
10.54.7 pg_Fetch_Array [Notes en ligne] [Exemples]	447
10.54.8 pg_Fetch_Object [Notes en ligne] [Exemples]	448
10.54.9 pg_Fetch_Row [Notes en ligne] [Exemples]	449
10.54.10 pg_FieldIsNull [Notes en ligne] [Exemples]	450
10.54.11 pg_FieldName [Notes en ligne] [Exemples]	450
10.54.12 pg_FieldNum [Notes en ligne] [Exemples]	450
10.54.13 pg_FieldPrtLen [Notes en ligne] [Exemples]	450
10.54.14 pg_FieldSize [Notes en ligne] [Exemples]	450
10.54.15 pg_FieldType [Notes en ligne] [Exemples]	451
10.54.16 pg_FreeResult [Notes en ligne] [Exemples]	451
10.54.17 pg_GetLastOid [Notes en ligne] [Exemples]	451
10.54.18 pg_Host [Notes en ligne] [Exemples]	451
10.54.19 pg_loclose [Notes en ligne] [Exemples]	451
10.54.20 pg_locreate [Notes en ligne] [Exemples]	451
10.54.21 pg_loopen [Notes en ligne] [Exemples]	452
10.54.22 pg_loread [Notes en ligne] [Exemples]	452
10.54.23 pg_loreadall [Notes en ligne] [Exemples]	452
10.54.24 pg_lounlink [Notes en ligne] [Exemples]	452
10.54.25 pg_lowrite [Notes en ligne] [Exemples]	452
10.54.26 pg_NumFields [Notes en ligne] [Exemples]	453
10.54.27 pg_NumRows [Notes en ligne] [Exemples]	453
10.54.28 pg_Options [Notes en ligne] [Exemples]	453
10.54.29 pg_pConnect [Notes en ligne] [Exemples]	453
10.54.30 pg_Port [Notes en ligne] [Exemples]	453
10.54.31 pg_Result [Notes en ligne] [Exemples]	454
10.54.32 pg_tty [Notes en ligne] [Exemples]	454
10.55 Fonctions POSIX [Notes en ligne]	454

Table of Contents

10.55.1 posix_kill [Notes en ligne] [Exemples]	454
10.55.2 posix_getpid [Notes en ligne] [Exemples]	454
10.55.3 posix_getppid [Notes en ligne] [Exemples]	455
10.55.4 posix_getuid [Notes en ligne] [Exemples]	455
10.55.5 posix_geteuid [Notes en ligne] [Exemples]	455
10.55.6 posix_getgid [Notes en ligne] [Exemples]	455
10.55.7 posix_getegid [Notes en ligne] [Exemples]	455
10.55.8 posix_setuid [Notes en ligne] [Exemples]	455
10.55.9 posix_setgid [Notes en ligne] [Exemples]	456
10.55.10 posix_getgroups [Notes en ligne] [Exemples]	456
10.55.11 posix_getlogin [Notes en ligne] [Exemples]	456
10.55.12 posix_getpgrp [Notes en ligne] [Exemples]	456
10.55.13 posix_setsid [Notes en ligne] [Exemples]	456
10.55.14 posix_setpgid [Notes en ligne] [Exemples]	456
10.55.15 posix_getpgid [Notes en ligne] [Exemples]	457
10.55.16 posix_getsid [Notes en ligne] [Exemples]	457
10.55.17 posix_uname [Notes en ligne] [Exemples]	457
10.55.18 posix_times [Notes en ligne] [Exemples]	457
10.55.19 posix_ctermid [Notes en ligne] [Exemples]	458
10.55.20 posix_ttyname [Notes en ligne] [Exemples]	458
10.55.21 posix_isatty [Notes en ligne] [Exemples]	458
10.55.22 posix_getcwd [Notes en ligne] [Exemples]	458
10.55.23 posix_mkfifo [Notes en ligne] [Exemples]	458
10.55.24 posix_getgrnam [Notes en ligne] [Exemples]	458
10.55.25 posix_getgrgid [Notes en ligne] [Exemples]	459
10.55.26 posix_getpwnam [Notes en ligne] [Exemples]	459
10.55.27 posix_getpwuid [Notes en ligne] [Exemples]	460
10.55.28 posix_getrlimit [Notes en ligne] [Exemples]	461
10.56 Pspell Functions [Notes en ligne]	461
10.56.1 pspell_new [Notes en ligne] [Exemples]	461
10.56.2 pspell_mode [Notes en ligne] [Exemples]	462
10.56.3 pspell_runtogether [Notes en ligne] [Exemples]	462
10.56.4 pspell_check [Notes en ligne] [Exemples]	462
10.56.5 pspell_suggest [Notes en ligne] [Exemples]	463
10.57 GNU Readline [Notes en ligne]	463
10.57.1 readline [Notes en ligne] [Exemples]	463
10.57.2 readline_add_history [Notes en ligne] [Exemples]	464
10.57.3 readline_clear_history [Notes en ligne] [Exemples]	464
10.57.4 readline_completion_function [Notes en ligne] [Exemples]	464
10.57.5 readline_info [Notes en ligne] [Exemples]	464
10.57.6 readline_list_history [Notes en ligne] [Exemples]	465
10.57.7 readline_read_history [Notes en ligne] [Exemples]	465
10.57.8 readline_write_history [Notes en ligne] [Exemples]	465
10.58 Fonction GNU Recode [Notes en ligne]	465
10.58.1 recode_string [Notes en ligne] [Exemples]	465
10.58.2 recode_file [Notes en ligne] [Exemples]	465
10.59 Expressions régulières [Notes en ligne]	466

Table of Contents

10.59.1	ereg [Notes en ligne] [Exemples].....	466
10.59.2	ereg_replace [Notes en ligne] [Exemples].....	467
10.59.3	eregi [Notes en ligne] [Exemples].....	468
10.59.4	eregi_replace [Notes en ligne] [Exemples].....	468
10.59.5	split [Notes en ligne] [Exemples].....	468
10.59.6	sql_regcase [Notes en ligne] [Exemples].....	469
10.60	Satellite CORBA client extension [Notes en ligne]	469
10.60.1	OrbitObject [Notes en ligne]	469
10.60.2	OrbitEnum [Notes en ligne]	470
10.60.3	OrbitStruct [Notes en ligne]	470
10.60.4	satellite_caught_exception [Notes en ligne] [Exemples].....	471
10.60.5	satellite_exception_id [Notes en ligne] [Exemples].....	472
10.60.6	satellite_exception_value [Notes en ligne] [Exemples].....	472
10.61	Sémaphores et gestion de la mémoire partagée [Notes en ligne]	472
10.61.1	sem_get [Notes en ligne] [Exemples].....	472
10.61.2	sem_acquire [Notes en ligne] [Exemples].....	473
10.61.3	sem_release [Notes en ligne] [Exemples].....	473
10.61.4	shm_attach [Notes en ligne] [Exemples].....	473
10.61.5	shm_detach [Notes en ligne] [Exemples].....	473
10.61.6	shm_remove [Notes en ligne] [Exemples].....	473
10.61.7	shm_put_var [Notes en ligne] [Exemples].....	474
10.61.8	shm_get_var [Notes en ligne] [Exemples].....	474
10.61.9	shm_remove_var [Notes en ligne] [Exemples].....	474
10.62	Gestion des sessions [Notes en ligne]	474
10.62.1	session_start [Notes en ligne] [Exemples].....	477
10.62.2	session_destroy [Notes en ligne] [Exemples].....	477
10.62.3	session_name [Notes en ligne] [Exemples].....	477
10.62.4	session_module_name [Notes en ligne] [Exemples].....	478
10.62.5	session_save_path [Notes en ligne] [Exemples].....	478
10.62.6	session_id [Notes en ligne] [Exemples].....	478
10.62.7	session_register [Notes en ligne] [Exemples].....	478
10.62.8	session_unregister [Notes en ligne] [Exemples].....	478
10.62.9	session_is_registered [Notes en ligne] [Exemples].....	479
10.62.10	session_decode [Notes en ligne] [Exemples].....	479
10.62.11	session_encode [Notes en ligne] [Exemples].....	479
10.63	Shared Memory Functions [Notes en ligne]	479
10.63.1	shmop_open [Notes en ligne] [Exemples].....	480
10.63.2	shmop_read [Notes en ligne] [Exemples].....	481
10.63.3	shmop_write [Notes en ligne] [Exemples].....	481
10.63.4	shmop_size [Notes en ligne] [Exemples].....	481
10.63.5	shmop_delete [Notes en ligne] [Exemples].....	482
10.63.6	shmop_close [Notes en ligne] [Exemples].....	482
10.64	SNMP functions [Notes en ligne]	482
10.64.1	snmpget [Notes en ligne] [Exemples].....	483
10.64.2	snmpset [Notes en ligne] [Exemples].....	483
10.64.3	snmpwalk [Notes en ligne] [Exemples].....	483
10.64.4	snmpwalkoid [Notes en ligne] [Exemples].....	484

Table of Contents

10.64.5 snmp_get_quick_print [Notes en ligne] [Exemples].....	484
10.64.6 snmp_set_quick_print [Notes en ligne] [Exemples].....	485
10.65 Socket functions [Notes en ligne]	485
10.65.1 accept_connect [Notes en ligne] [Exemples].....	487
10.65.2 bind [Notes en ligne] [Exemples].....	488
10.65.3 connect [Notes en ligne] [Exemples].....	488
10.65.4 listen [Notes en ligne] [Exemples].....	488
10.65.5 socket [Notes en ligne] [Exemples].....	489
10.65.6 strerror [Notes en ligne] [Exemples].....	489
10.66 Fonctions de chaîne de caractères [Notes en ligne]	489
10.66.1 AddCSlashes [Notes en ligne] [Exemples].....	490
10.66.2 AddSlashes [Notes en ligne] [Exemples].....	490
10.66.3 bin2hex [Notes en ligne] [Exemples].....	490
10.66.4 Chop [Notes en ligne] [Exemples].....	490
10.66.5 Chr [Notes en ligne] [Exemples].....	491
10.66.6 chunk_split [Notes en ligne] [Exemples].....	491
10.66.7 convert_cyr_string [Notes en ligne] [Exemples].....	491
10.66.8 count_chars [Notes en ligne] [Exemples].....	492
10.66.9 crypt [Notes en ligne] [Exemples].....	492
10.66.10 echo [Notes en ligne] [Exemples].....	493
10.66.11 explode [Notes en ligne] [Exemples].....	493
10.66.12 get_html_translation_table [Notes en ligne] [Exemples].....	493
10.66.13 get_meta_tags [Notes en ligne] [Exemples].....	494
10.66.14 htmlentities [Notes en ligne] [Exemples].....	494
10.66.15 htmlspecialchars [Notes en ligne] [Exemples].....	494
10.66.16 implode [Notes en ligne] [Exemples].....	495
10.66.17 join [Notes en ligne] [Exemples].....	495
10.66.18 ltrim [Notes en ligne] [Exemples].....	495
10.66.19 md5 [Notes en ligne] [Exemples].....	495
10.66.20 Metaphone [Notes en ligne] [Exemples].....	496
10.66.21 nl2br [Notes en ligne] [Exemples].....	496
10.66.22 Ord [Notes en ligne] [Exemples].....	496
10.66.23 parse_str [Notes en ligne] [Exemples].....	496
10.66.24 print [Notes en ligne] [Exemples].....	497
10.66.25 printf [Notes en ligne] [Exemples].....	497
10.66.26 quoted_printable_decode [Notes en ligne] [Exemples].....	497
10.66.27 QuoteMeta [Notes en ligne] [Exemples].....	497
10.66.28 rawurldecode [Notes en ligne] [Exemples].....	498
10.66.29 rawurlencode [Notes en ligne] [Exemples].....	498
10.66.30 setlocale [Notes en ligne] [Exemples].....	498
10.66.31 similar_text [Notes en ligne] [Exemples].....	499
10.66.32 soundex [Notes en ligne] [Exemples].....	499
10.66.33 sprintf [Notes en ligne] [Exemples].....	499
10.66.34 strcasecmp [Notes en ligne] [Exemples].....	501
10.66.35 strchr [Notes en ligne] [Exemples].....	501
10.66.36 strcmp [Notes en ligne] [Exemples].....	501
10.66.37 strcspn [Notes en ligne] [Exemples].....	501

Table of Contents

10.66.38 strip_tags [Notes en ligne] [Exemples]	501
10.66.39 StripCSlashes [Notes en ligne] [Exemples]	502
10.66.40 StripSlashes [Notes en ligne] [Exemples]	502
10.66.41 stristr [Notes en ligne] [Exemples]	502
10.66.42 strlen [Notes en ligne] [Exemples]	502
10.66.43 strpos [Notes en ligne] [Exemples]	503
10.66.44 strrchr [Notes en ligne] [Exemples]	503
10.66.45 str_repeat [Notes en ligne] [Exemples]	504
10.66.46 strrev [Notes en ligne] [Exemples]	504
10.66.47 strrpos [Notes en ligne] [Exemples]	504
10.66.48 strspn [Notes en ligne] [Exemples]	504
10.66.49 strstr [Notes en ligne] [Exemples]	504
10.66.50 strtok [Notes en ligne] [Exemples]	505
10.66.51 strtolower [Notes en ligne] [Exemples]	505
10.66.52 strtoupper [Notes en ligne] [Exemples]	506
10.66.53 str_replace [Notes en ligne] [Exemples]	506
10.66.54 strstr [Notes en ligne] [Exemples]	506
10.66.55 substr [Notes en ligne] [Exemples]	507
10.66.56 substr_replace [Notes en ligne] [Exemples]	508
10.66.57 trim [Notes en ligne] [Exemples]	508
10.66.58 ucfirst [Notes en ligne] [Exemples]	508
10.66.59 ucwords [Notes en ligne] [Exemples]	509
10.67 Fonctions Shockwave Flash [Notes en ligne]	509
10.67.1 swf_openfile [Notes en ligne] [Exemples]	510
10.67.2 swf_closefile [Notes en ligne] [Exemples]	510
10.67.3 swf_labelframe [Notes en ligne] [Exemples]	510
10.67.4 swf_showframe [Notes en ligne] [Exemples]	511
10.67.5 swf_setframe [Notes en ligne] [Exemples]	511
10.67.6 swf_getframe [Notes en ligne] [Exemples]	511
10.67.7 swf_mulcolor [Notes en ligne] [Exemples]	511
10.67.8 swf_addcolor [Notes en ligne] [Exemples]	511
10.67.9 swf_placeobject [Notes en ligne] [Exemples]	511
10.67.10 swf_modifyobject [Notes en ligne] [Exemples]	512
10.67.11 swf_removeobject [Notes en ligne] [Exemples]	512
10.67.12 swf_nextid [Notes en ligne] [Exemples]	512
10.67.13 swf_startdoaction [Notes en ligne] [Exemples]	512
10.67.14 swf_actiongotoframe [Notes en ligne] [Exemples]	512
10.67.15 swf_actiongeturl [Notes en ligne] [Exemples]	513
10.67.16 swf_actionnextframe [Notes en ligne] [Exemples]	513
10.67.17 swf_actionprevframe [Notes en ligne] [Exemples]	513
10.67.18 swf_actionplay [Notes en ligne] [Exemples]	513
10.67.19 swf_actionstop [Notes en ligne] [Exemples]	513
10.67.20 swf_actiontogglequality [Notes en ligne] [Exemples]	513
10.67.21 swf_actionwaitforframe [Notes en ligne] [Exemples]	514
10.67.22 swf_actionsettarget [Notes en ligne] [Exemples]	514
10.67.23 swf_actiongotolabel [Notes en ligne] [Exemples]	514
10.67.24 swf_enddoaction [Notes en ligne] [Exemples]	514

Table of Contents

10.67.25 swf_defineline [Notes en ligne] [Exemples].....	514
10.67.26 swf_definerect [Notes en ligne] [Exemples].....	514
10.67.27 swf_definepoly [Notes en ligne] [Exemples].....	515
10.67.28 swf_startshape [Notes en ligne] [Exemples].....	515
10.67.29 swf_shapelinesolid [Notes en ligne] [Exemples].....	515
10.67.30 swf_shapefilloff [Notes en ligne] [Exemples].....	515
10.67.31 swf_shapefillsolid [Notes en ligne] [Exemples].....	515
10.67.32 swf_shapefillbitmapclip [Notes en ligne] [Exemples].....	515
10.67.33 swf_shapefillbitmaptile [Notes en ligne] [Exemples].....	516
10.67.34 swf_shapemoveto [Notes en ligne] [Exemples].....	516
10.67.35 swf_shapelineto [Notes en ligne] [Exemples].....	516
10.67.36 swf_shapecurveto [Notes en ligne] [Exemples].....	516
10.67.37 swf_shapecurveto3 [Notes en ligne] [Exemples].....	516
10.67.38 swf_shapearc [Notes en ligne] [Exemples].....	516
10.67.39 swf_endshape [Notes en ligne] [Exemples].....	517
10.67.40 swf_definefont [Notes en ligne] [Exemples].....	517
10.67.41 swf_setfont [Notes en ligne] [Exemples].....	517
10.67.42 swf_fontsize [Notes en ligne] [Exemples].....	517
10.67.43 swf_fontslant [Notes en ligne] [Exemples].....	517
10.67.44 swf_fontracking [Notes en ligne] [Exemples].....	517
10.67.45 swf_getfontinfo [Notes en ligne] [Exemples].....	518
10.67.46 swf_definetext [Notes en ligne] [Exemples].....	518
10.67.47 swf_textwidth [Notes en ligne] [Exemples].....	518
10.67.48 swf_definebitmap [Notes en ligne] [Exemples].....	518
10.67.49 swf_getbitmapinfo [Notes en ligne] [Exemples].....	518
10.67.50 swf_startsymbol [Notes en ligne] [Exemples].....	519
10.67.51 swf_endsymbol [Notes en ligne] [Exemples].....	519
10.67.52 swf_startbutton [Notes en ligne] [Exemples].....	519
10.67.53 swf_addbuttonrecord [Notes en ligne] [Exemples].....	519
10.67.54 swf_oncondition [Notes en ligne] [Exemples].....	520
10.67.55 swf_endbutton [Notes en ligne] [Exemples].....	520
10.67.56 swf_viewport [Notes en ligne] [Exemples].....	520
10.67.57 swf_ortho [Notes en ligne] [Exemples].....	521
10.67.58 swf_ortho2 [Notes en ligne] [Exemples].....	521
10.67.59 swf_perspective [Notes en ligne] [Exemples].....	521
10.67.60 swf_polarview [Notes en ligne] [Exemples].....	521
10.67.61 swf_lookat [Notes en ligne] [Exemples].....	521
10.67.62 swf_pushmatrix [Notes en ligne] [Exemples].....	522
10.67.63 swf_popmatrix [Notes en ligne] [Exemples].....	522
10.67.64 swf_scale [Notes en ligne] [Exemples].....	522
10.67.65 swf_translate [Notes en ligne] [Exemples].....	522
10.67.66 swf_rotate [Notes en ligne] [Exemples].....	522
10.67.67 swf_posround [Notes en ligne] [Exemples].....	522
10.68 Sybase [Notes en ligne]	523
10.68.1 sybase_affected_rows [Notes en ligne] [Exemples].....	523
10.68.2 sybase_close [Notes en ligne] [Exemples].....	523
10.68.3 sybase_connect [Notes en ligne] [Exemples].....	523

Table of Contents

10.68.4 sybase_data_seek [Notes en ligne] [Exemples].....	524
10.68.5 sybase_fetch_array [Notes en ligne] [Exemples].....	524
10.68.6 sybase_fetch_field [Notes en ligne] [Exemples].....	524
10.68.7 sybase_fetch_object [Notes en ligne] [Exemples].....	524
10.68.8 sybase_fetch_row [Notes en ligne] [Exemples].....	525
10.68.9 sybase_field_seek [Notes en ligne] [Exemples].....	525
10.68.10 sybase_free_result [Notes en ligne] [Exemples].....	525
10.68.11 sybase_get_last_message [Notes en ligne] [Exemples].....	525
10.68.12 sybase_min_client_severity [Notes en ligne] [Exemples].....	526
10.68.13 sybase_min_error_severity [Notes en ligne] [Exemples].....	526
10.68.14 sybase_min_message_severity [Notes en ligne] [Exemples].....	526
10.68.15 sybase_min_server_severity [Notes en ligne] [Exemples].....	526
10.68.16 sybase_num_fields [Notes en ligne] [Exemples].....	526
10.68.17 sybase_num_rows [Notes en ligne] [Exemples].....	527
10.68.18 sybase_pconnect [Notes en ligne] [Exemples].....	527
10.68.19 sybase_query [Notes en ligne] [Exemples].....	527
10.68.20 sybase_result [Notes en ligne] [Exemples].....	527
10.68.21 sybase_select_db [Notes en ligne] [Exemples].....	528
10.69 ODBC [Notes en ligne]	528
10.69.1 odbc_autocommit [Notes en ligne] [Exemples].....	528
10.69.2 odbc_binmode [Notes en ligne] [Exemples].....	528
10.69.3 odbc_close [Notes en ligne] [Exemples].....	529
10.69.4 odbc_close_all [Notes en ligne] [Exemples].....	529
10.69.5 odbc_commit [Notes en ligne] [Exemples].....	529
10.69.6 odbc_connect [Notes en ligne] [Exemples].....	529
10.69.7 odbc_cursor [Notes en ligne] [Exemples].....	530
10.69.8 odbc_do [Notes en ligne] [Exemples].....	530
10.69.9 odbc_exec [Notes en ligne] [Exemples].....	530
10.69.10 odbc_execute [Notes en ligne] [Exemples].....	531
10.69.11 odbc_fetch_into [Notes en ligne] [Exemples].....	531
10.69.12 odbc_fetch_row [Notes en ligne] [Exemples].....	531
10.69.13 odbc_field_name [Notes en ligne] [Exemples].....	531
10.69.14 odbc_field_type [Notes en ligne] [Exemples].....	531
10.69.15 odbc_field_len [Notes en ligne] [Exemples].....	532
10.69.16 odbc_free_result [Notes en ligne] [Exemples].....	532
10.69.17 odbc_longreadlen [Notes en ligne] [Exemples].....	532
10.69.18 odbc_num_fields [Notes en ligne] [Exemples].....	532
10.69.19 odbc_pconnect [Notes en ligne] [Exemples].....	532
10.69.20 odbc_prepare [Notes en ligne] [Exemples].....	533
10.69.21 odbc_num_rows [Notes en ligne] [Exemples].....	533
10.69.22 odbc_result [Notes en ligne] [Exemples].....	533
10.69.23 odbc_result_all [Notes en ligne] [Exemples].....	534
10.69.24 odbc_rollback [Notes en ligne] [Exemples].....	534
10.69.25 odbc_setoption [Notes en ligne] [Exemples].....	534
10.70 Fonctions URL [Notes en ligne]	535
10.70.1 base64_decode [Notes en ligne] [Exemples].....	535
10.70.2 base64_encode [Notes en ligne] [Exemples].....	535

Table of Contents

10.70.3	parse_url [Notes en ligne] [Exemples]	535
10.70.4	urldecode [Notes en ligne] [Exemples]	535
10.70.5	urlencode [Notes en ligne] [Exemples]	536
10.71	Fonctions sur les variables [Notes en ligne]	536
10.71.1	doubleval [Notes en ligne] [Exemples]	536
10.71.2	empty [Notes en ligne] [Exemples]	537
10.71.3	gettype [Notes en ligne] [Exemples]	537
10.71.4	intval [Notes en ligne] [Exemples]	537
10.71.5	is_array [Notes en ligne] [Exemples]	538
10.71.6	is_double [Notes en ligne] [Exemples]	538
10.71.7	is_float [Notes en ligne] [Exemples]	538
10.71.8	is_int [Notes en ligne] [Exemples]	538
10.71.9	is_integer [Notes en ligne] [Exemples]	538
10.71.10	is_long [Notes en ligne] [Exemples]	538
10.71.11	is_object [Notes en ligne] [Exemples]	539
10.71.12	is_real [Notes en ligne] [Exemples]	539
10.71.13	is_string [Notes en ligne] [Exemples]	539
10.71.14	isset [Notes en ligne] [Exemples]	539
10.71.15	print_r [Notes en ligne] [Exemples]	539
10.71.16	settype [Notes en ligne] [Exemples]	540
10.71.17	strval [Notes en ligne] [Exemples]	540
10.71.18	unset [Notes en ligne] [Exemples]	540
10.71.19	var_dump [Notes en ligne] [Exemples]	541
10.72	Fonctions Vmailmgr [Notes en ligne]	541
10.72.1	vm_adduser [Notes en ligne] [Exemples]	542
10.72.2	vm_addalias [Notes en ligne] [Exemples]	542
10.72.3	vm_passwd [Notes en ligne] [Exemples]	542
10.72.4	vm_delalias [Notes en ligne] [Exemples]	542
10.72.5	vm_deluser [Notes en ligne] [Exemples]	542
10.73	WDDX functions [Notes en ligne]	542
10.73.1	wddx_serialize_value [Notes en ligne] [Exemples]	543
10.73.2	wddx_serialize_vars [Notes en ligne] [Exemples]	543
10.73.3	wddx_packet_start [Notes en ligne] [Exemples]	544
10.73.4	wddx_packet_end [Notes en ligne] [Exemples]	544
10.73.5	wddx_add_vars [Notes en ligne] [Exemples]	544
10.73.6	wddx_deserialize [Notes en ligne] [Exemples]	545
10.74	Analyseur syntaxique XML [Notes en ligne]	545
10.74.1	Introduction [Notes en ligne]	545
10.74.1.1	A propos de XML [Notes en ligne]	545
10.74.1.2	Installation [Notes en ligne]	545
10.74.1.3	A propos de cette extension : [Notes en ligne]	545
10.74.1.4	Problèmes de casse [Notes en ligne]	546
10.74.1.5	Error Codes [Notes en ligne]	547
10.74.1.6	Codage des caractères [Notes en ligne]	547
10.74.2	Quelques exemples [Notes en ligne]	548
10.74.2.1	Exemple de structure XML [Notes en ligne]	548
10.74.2.2	XML Transtypage XML -> HTML [Notes en ligne]	548

Table of Contents

10.74.2.3	XML Entité externe [Notes en ligne]	549
10.74.3	xml_parser_create [Notes en ligne] [Exemples]	552
10.74.4	xml_set_object [Notes en ligne] [Exemples]	552
10.74.5	xml_set_element_handler [Notes en ligne] [Exemples]	553
10.74.6	xml_set_character_data_handler [Notes en ligne] [Exemples]	554
10.74.7	xml_set_processing_instruction_handler [Notes en ligne] [Exemples]	554
10.74.8	xml_set_default_handler [Notes en ligne] [Exemples]	555
10.74.9	xml_set_unparsed_entity_decl_handler [Notes en ligne] [Exemples]	555
10.74.10	xml_set_notation_decl_handler [Notes en ligne] [Exemples]	556
10.74.11	xml_set_external_entity_ref_handler [Notes en ligne] [Exemples]	557
10.74.12	xml_parse [Notes en ligne] [Exemples]	558
10.74.13	xml_get_error_code [Notes en ligne] [Exemples]	558
10.74.14	xml_error_string [Notes en ligne] [Exemples]	558
10.74.15	xml_get_current_line_number [Notes en ligne] [Exemples]	559
10.74.16	xml_get_current_column_number [Notes en ligne] [Exemples]	559
10.74.17	xml_get_current_byte_index [Notes en ligne] [Exemples]	559
10.74.18	xml_parser_free [Notes en ligne] [Exemples]	559
10.74.19	xml_parser_set_option [Notes en ligne] [Exemples]	560
10.74.20	xml_parser_get_option [Notes en ligne] [Exemples]	560
10.74.21	utf8_decode [Notes en ligne] [Exemples]	561
10.74.22	utf8_encode [Notes en ligne] [Exemples]	561
10.75	YAZ [Notes en ligne]	561
10.75.1	yaz_addinfo [Notes en ligne] [Exemples]	563
10.75.2	yaz_close [Notes en ligne] [Exemples]	563
10.75.3	yaz_connect [Notes en ligne] [Exemples]	563
10.75.4	yaz_errno [Notes en ligne] [Exemples]	563
10.75.5	yaz_error [Notes en ligne] [Exemples]	564
10.75.6	yaz_hits [Notes en ligne] [Exemples]	564
10.75.7	yaz_range [Notes en ligne] [Exemples]	564
10.75.8	yaz_record [Notes en ligne] [Exemples]	564
10.75.9	yaz_search [Notes en ligne] [Exemples]	564
10.75.10	yaz_syntax [Notes en ligne] [Exemples]	565
10.75.11	yaz_wait [Notes en ligne] [Exemples]	565
10.76	Compression [Notes en ligne]	565
10.76.1	Petit exemple [Notes en ligne]	565
10.76.2	gzclose [Notes en ligne] [Exemples]	566
10.76.3	gzeof [Notes en ligne] [Exemples]	566
10.76.4	gzfile [Notes en ligne] [Exemples]	566
10.76.5	gzgetc [Notes en ligne] [Exemples]	566
10.76.6	gzgets [Notes en ligne] [Exemples]	566
10.76.7	gzgetss [Notes en ligne] [Exemples]	567
10.76.8	gzopen [Notes en ligne] [Exemples]	567
10.76.9	gzpassthru [Notes en ligne] [Exemples]	567
10.76.10	gzputs [Notes en ligne] [Exemples]	568
10.76.11	gzread [Notes en ligne] [Exemples]	568
10.76.12	gzrewind [Notes en ligne] [Exemples]	568
10.76.13	gzseek [Notes en ligne] [Exemples]	568

Table of Contents

10.76.14 gztell [Notes en ligne] [Exemples]	569
10.76.15 gzwrite [Notes en ligne] [Exemples]	569
10.76.16 readgzfile [Notes en ligne] [Exemples]	569
A Débugueur PHP [Notes en ligne]	570
A.1 Utiliser le débugeur PHP [Notes en ligne]	570
A.2 Protocole du débugeur [Notes en ligne]	570
B Migration de PHP/FI 2.0 à PHP 3.0 [Notes en ligne]	573
B.1 A propos des incompatibilités en 3.0 [Notes en ligne]	573
B.2 Start/end tags [Notes en ligne]	573
B.3 if..endif syntax [Notes en ligne]	574
B.4 while syntax [Notes en ligne]	574
B.5 Types d'expression [Notes en ligne]	575
B.6 Les messages d'erreur ont changé [Notes en ligne]	575
B.7 Evaluation rapide des booléens [Notes en ligne]	576
B.8 La valeur true/false comme retour de fonctions [Notes en ligne]	576
B.9 Diverses incompatibilités [Notes en ligne]	576
C Développement PHP [Notes en ligne]	578
C.1 Adding functions to PHP3 [Notes en ligne]	578
C.1.1 Prototypes de fonctions [Notes en ligne]	578
C.1.2 Arguments de fonctions [Notes en ligne]	578
C.1.3 Fonctions à nombre d'arguments variable [Notes en ligne]	578
C.1.4 Utiliser les arguments d'une fonction [Notes en ligne]	579
C.1.5 Gestion de la mémoire dans une fonction [Notes en ligne]	580
C.1.6 Affecter une variable dans la table des symboles [Notes en ligne]	580
C.1.7 Retourne une valeur simple [Notes en ligne]	582
C.1.8 Retourner des valeurs complexes [Notes en ligne]	582
C.1.9 Using the resource list [Notes en ligne]	583
C.1.10 Utiliesr la table des ressources persistantes. [Notes en ligne]	584
C.1.11 Ajouter des directives de configuration à l'exécution [Notes en ligne]	586
C.2 Appeler des fonctions utilisateurs [Notes en ligne]	586
C.2.1 HashTable *function table [Notes en ligne]	586
C.2.2 pval *object [Notes en ligne]	586
C.2.3 pval *function_name [Notes en ligne]	587
C.2.4 pval *retval [Notes en ligne]	587
C.2.5 int param count [Notes en ligne]	587
C.2.6 pval *params[] [Notes en ligne]	587
C.3 Rapport d'erreurs [Notes en ligne]	587
C.3.1 E NOTICE [Notes en ligne]	587
C.3.2 E WARNING [Notes en ligne]	588
C.3.3 E ERROR [Notes en ligne]	588
C.3.4 E PARSE [Notes en ligne]	588
C.3.5 E CORE_ERROR [Notes en ligne]	588
C.3.6 E CORE_WARNING [Notes en ligne]	588
C.3.7 E_USER_ERROR [Notes en ligne]	588

Table of Contents

C.3.8 E USER WARNING [Notes en ligne]	588
C.3.9 E USER NOTICE [Notes en ligne]	589
Index des fonctions [Notes en ligne] [Exemples].....	590
A	590
B	591
C	591
D	593
E	594
F	595
G	596
H	597
I	599
J	602
K	602
L	603
M	603
N	607
O	607
P	609
Q	612
R	613
S	613
T	617
U	617
V	617
W	618
X	618
Y	618
Z	619
Index des concepts [Notes en ligne].....	620
A	620
B	622
C	622
D	625
E	627
F	629
G	632
H	632
I	632
J	633
L	633
M	635
N	636
O	636
P	637

Table of Contents

R	638
S	645
T	646
U	647
V	647
W	647
Index des exemples [Notes en ligne].....	648
A	648
B	648
C	648
D	649
E	649
F	652
G	652
H	652
I	653
L	653
M	653
N	653
O	653
P	654
R	654
S	654
T	655
U	655
V	655
W	655
X	655

1 Préface

[\[Notes en ligne\]](#)

PHP, signifie "PHP: Hypertext Preprocessor" (Preprocesseur HyperTexte) , est un langage de script HTML. La plupart de sa syntaxe est empruntée au langages C, Java et Perl, mais y ajoute plusieurs fonctionnalités uniques. Le but de ce langage est de permettre aux développeurs web de concevoir rapidement des sites, aux pages dynamiques

1.1 A propos de ce manuel

[\[Notes en ligne\]](#)

Ce manuel est écrit en *SGML* en utilisant [DocBook DTD](#), avec [DSSSL](#) (Document Style and Semantics Specification Language) pour le formattage. Les utilitaires utilisés pour générer le format *HTML*, *TeX* et *RTF* sont [Jade](#), écrit [James Clark](#) et [The Modular DocBook Stylesheets](#) écrit par [Norman Walsh](#). La documentation PHP a été assemblée par [texi](#).

Le manuel est mis à jour tous les jours, y compris les traductions. Il est disponible à l'adresse <http://snaps.php.net/manual/>. Ce manuel a été traduit en Français par dams@php.net Il a été généré à partir de la documentation en Anglais originale du PHP documentation Group, au format XML, gr'ce à une version adaptée de [texi](#).

2 Preface

[\[Notes en ligne\]](#)

2.1 Les auteurs

[\[Notes en ligne\]](#)

Stig Sæther Bakken Alexander Aulbach Egon Schmid Jim Winstead Lars Torben Wilson Rasmus Lerdorf
Zeev Suraski Andrei Zmievski Damien Seguy 1997 1998 1999 2000 *PHP Documentation Group*

2.2 Copyright

[\[Notes en ligne\]](#)

Ce manuel est © Copyright 1997, 1998, 1999, 2000 par PHP Documentation Group. Les membres de ce groupe sont listés [2.1 Les auteurs](#).

Ce manuel peut être redistribué sous licence GNU General Public License, comme stipulé par la Free Software Foundation; soit la version 2 de la Licence, soit (à votre choix), une version ultérieure.

3 Copyright, distribution, historique

[\[Notes en ligne\]](#)

PHP est copyright (C) 1997 par PHP Development Team. Les membres de cette équipe sont listés dans le fichier de crédits, fourni avec la distribution source de PHP 3.0.

PHP est un logiciel libre : vous pouvez le modifier et/ou le modifier sous licence GNU General Public License, telle que publiée par Free Software Foundation; utilisez soit la version 2 de la License, ou toute version ultérieure (à votre convenance).

PHP est distribué avec l'espoir qu'il sera utile, mais il l'est SANS AUCUNE GARANTIE; sans même la garantie de COMMERCIALISATION ou d'UTILITE POUR UN BUT QUELCONQUE. Reportez vous à la licence GNU General Public License pour plus de détails.

Vous devez avoir reçu une copie de la GNU General Public License avec ce programme; sinon, écrivez à la fondation Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

4 Installation

[\[Notes en ligne\]](#)

4.1 Télécharger la dernière version

[\[Notes en ligne\]](#)

Le code source ainsi que des binaires pour certaines plates-formes (notamment Windows), sont disponibles à l'adresse suivante: <http://www.php.net/>.

4.2 Installation sous UNIX

[\[Notes en ligne\]](#)

Ce chapitre va vous aider lors de la configuration et de l'installation du PHP. Les connaissances requises sont les suivantes :

- Connaissances basiques d'UNIX (savoir faire un "make" et utiliser un compilateur C)
- Avoir un compilateur C ANSI installé
- Avoir installé un serveur web

Il y a plusieurs façons de compiler et configurer PHP pour une plate-forme UNIX. Le processus de compilation est contrôlé entièrement par les options de ligne de commande du script `configure`. Cette documentation souligne les options les plus fréquentes, mais il en existe un grand nombre. Jetez un oeil à [4.3 Liste complète des options de configuration](#) pour une documentation exhaustive.

- [4.2.1 Installation rapide \(Version Module Apache\)](#)
- [4.2.2 Module fhttpd](#)
- Pour l'utiliser avec [4.2.3 Autres serveurs web](#)
- Sous forme d' [4.2.4 `~/usr/local/src/fhttpd`](#)

4.2.1 Installation rapide (Version Module Apache)

[\[Notes en ligne\]](#)

PHP peut être compilé de nombreuses manières différentes pour en faire un module Apache. Voyons d'abord les instructions rapides, puis une liste d'exemples divers, avec les explications. Comme cela nous aurons survolé l'ensemble de l'installation.

Vous pouvez sélectionner les arguments à ajouter à la commande `configure` (ligne 8, ci dessous) parmi la liste [4.3 Liste complète des options de configuration](#).

Instructions d'installation rapide (Version Module Apache) *Instructions d'installation rapide (Version Module Apache)*

1. `gunzip apache_1.3.x.tar.gz`
2. `tar xvf apache_1.3.x.tar`

```

3. gunzip php-3.0.x.tar.gz
4. tar xvf php-3.0.x.tar
5. cd apache_1.3.x
6. ./configure --prefix=/www
7. cd ../php-3.0.x
8. ./configure --with-mysql --with-apache=../apache_1.3.x --enable-track-vars
9. make
10. make install
11. cd ../apache_1.3.x
12. ./configure --prefix=/www --activate-module=src/modules/php3/libphp3.a
13. make
14. make install

```

A la place de cette étape, vous pouvez simplement écraser le binaire httpd. Assurez-vous d'avoir bien arrêté le démon d'abord.

```

15. cd ../php-3.0.x
16. cp php3.ini-dist /usr/local/lib/php3.ini

```

Vous pouvez éditer le fichier de configuration /usr/local/lib/php3.ini. Si vous préférez installer le fichier dans un autre répertoire, il faut utiliser l'option de configuration --with-config-file-path=/path à l'étape 8.

```

17. Editez le fichier de configuration apache httpd.conf ou srm.conf et ajoutez :
AddType application/x-httpd-php3 .php3

```

Ici, il faut choisir l'extension que vous souhaitez donner au fichier php. .php est simplement celle que nous suggérons.

```

18. Utilisez la procédure normale afin de démarrer le serveur Apache. (Vous devez impérativement arrêter et redémarrer le serveur Apache, et pas seulement le relancer à l'aide d'un signal HUP ou USR1).

```

```
./configure --with-apxs --with-pgsql
```

Cette option va créer un fichier `libphp4.so`, qui est une bibliothèque partagée, chargée par Apache grâce à la ligne `LoadModule` dans le fichier de configuration d'Apache `httpd.conf`. Le support de la base de données PostgreSQL est compris dans la bibliothèque `libphp4.so`.

```
./configure --with-apxs --with-pgsql=shared
```

Cette option va créer un fichier `libphp4.so` qui est une bibliothèque partagée Apache, mais elle va aussi créer une bibliothèque partagée `pgsql.so` qui sera chargée par PHP soit avec les directives de chargement du fichier de configuration `php.ini` ou avec la fonction de chargement [dl\(\)](#).

```
./configure --with-apache=/path/to/apache_source --with-pgsql
```

Cette option va créer la bibliothèque `libmodphp4.a`, le fichier `mod_php4.c` et quelques autres fichiers utilitaires, puis copier tout cela dans le dossier `src/modules/php4` du dossier source Apache. Lorsque vous compilez Apache avec l'option `--activate-module=src/modules/php4/libphp4.a`, le compilateur d'Apache créera le fichier `libphp4.a` et le liera statiquement avec `httpd`. Le support PostgreSQL est inclus directement dans l'exécutable `httpd`, ce qui fait que le résultat est un fichier exécutable unique `httpd`, qui combine Apache et tout PHP.

```
./configure --with-apache=/path/to/apache_source --with-pgsql=shared
```

Même chose que le précédent, mais au lieu d'inclure le support de PostgreSQL directement dans le `httpd` final, vous obtiendrez une bibliothèque partagée `pgsql.so` que vous pourrez charger dans PHP,

grâce au fichier de configuration `php.ini` ou la fonction [dl\(\)](#).

Lors du choix de compilation de PHP, prenez bien en considération les avantages et les inconvénients de chaque méthode. Compiler PHP comme une bibliothèque partagée vous permet de compiler séparément Apache et toutes les extensions PHP. Compiler PHP statiquement vous donne une plus grande vitesse d'exécution. Pour plus de détails reportez vous à la documentation Apache [support DSO](#) (en anglais).

4.2.2 Module fhttpd

[\[Notes en ligne\]](#)

Pour compiler PHP comme un module fhttpd, répondre "yes" à la question "Build as an fhttpd module ?" (cela correspond à l'option de configuration `@xref{with-fhttpd}=DIR` et spécifier la racine de la distribution fhttpd. Le répertoire par défaut est: `~/usr/local/src/fhttpd`. Si vous utilisez fhttpd, compiler PHP en module vous permettra d'obtenir des performances supérieures, plus de contrôle et la possibilité d'exécution à distance.

4.2.3 Autres serveurs web

[\[Notes en ligne\]](#)

PHP peut être compilé pour fonctionner avec de nombreux autres serveurs web. Reportez vous à [4.3.7 Serveur](#) pour une liste complète des options de configuration.

4.2.4 ~/usr/local/src/fhttpd'

[\[Notes en ligne\]](#)

Par défaut, PHP est compilé comme une CGI. Si vous voulez que votre serveur web supporte le PHP, compiler le PHP comme une CGI permet d'obtenir de meilleures performances. Cependant, la version CGI permet aux utilisateurs de lancer des scripts PHP sous leur UID respectives. Lisez attentivement le chapitre consacré à la [6 Sécurité](#) si vous souhaitez utiliser cette solution.

4.2.5 Options de base de données

[\[Notes en ligne\]](#)

PHP dispose du support natif d'un grand nombre de base de données et d'ODBC. Pour activer les différentes bases de données, vous pouvez utiliser les options du script `configure`, au moment de la compilation.

Lisez la [4.3.1 Configuration pour le support des bases de données](#) pour plus de détails.

La liste complète des options de `configure`, reportez vous à la [4.3 Liste complète des options de configuration](#).

4.2.6 Compilation

[\[Notes en ligne\]](#)

Lorsque PHP est configuré, vous êtes prêts à compiler un exécutable CGI, ou une bibliothèque HP. La commande `make` devrait faire le travail. Si ce n'est pas le cas, et que vous comprenez pas pourquoi, allez à [4.5 Problèmes?](#)

[4.2.7 Tests](#)

[\[Notes en ligne\]](#)

Si vous avez compilé PHP comme programme CGI, vous pouvez tester votre produit en tapant : `make test`. C'est toujours une bonne chose de tester le résultat d'une compilation. Cela vous permet de repérer des problèmes entre PHP et votre plate-forme, bien plus facilement que si vous attendez.

[4.2.8 Performances](#)

[\[Notes en ligne\]](#)

Si vous avez compilé PHP comme programme CGI, vous pouvez évaluer les performances de PHP avec la commande `make bench`. Notez que si le "safe mode" est activé (par défaut), vous ne risquez de voir l'évaluation s'arrêter une fois les 30 secondes réglementaires écoulées. En effet, la fonction `set_time_limit()` ne peut pas être utilisé si le "safe mode" fonctionne. Utilisez l'option [7.1.1.19 ini.max-execution-time](#) pour contrôler le temps d'exécutions de vos scripts. `make bench` ignore le fichier de [7.1 Le fichier de configuration](#).

[4.3 Liste complète des options de configuration](#)

[\[Notes en ligne\]](#)

Note : *Ces options ne sont utilisées que lors de la compilation. Si vous voulez modifier le comportement de PHP au moment de l'exécution, reportez vous à la partie [7 Configuration](#).*

Ceci est la liste complète des options de configurations supportées par le script de configuration PHP 3 et PHP 4 ``configure'`, utilisé lors de la compilation en environnement UNIX. Certaines sont disponibles sous PHP 3, d'autres sous PHP 4, d'autres encore pour les deux versions. De nombreuses options ont changé de nom entre PHP 3 et PHP 4, mais font exactement la même chose. Les entrées disposent d'un lien entre elles (si vous avez un problème avec une option PHP 3, vérifier ainsi que les noms n'ont pas changé).

- [4.3.1 Configuration pour le support des bases de données](#)
- [4.3.2 Ecommerce](#)
- [4.3.3 Graphisme](#)
- [4.3.4 Divers](#)
- [4.3.5 Réseau](#)
- [4.3.6 Configuration de PHP](#)
- [4.3.7 Serveur](#)
- [4.3.8 Texte et langue](#)
- [4.3.9 XML](#)

[4.3.1 Configuration pour le support des bases de données](#)

[\[Notes en ligne\]](#)

PHP supporte de nombreuses bases de données (et aussi ODBC):

Pour la liste détaillée des options de ``configure'`, reportez vous à [4.3 Liste complète des options de configuration](#).

[4.3.1.1 install.configure.with-adabas](#)

[\[Notes en ligne\]](#)

--with-adabas[=DIR]

- PHP 3, PHP 4: Ajoute le support Adabas. DIR est le dossier de l'installation Adabas (par défaut : /usr/local).

[Site d'Adabas](#)

[4.3.1.2 install.configure.enable-dba](#)

[\[Notes en ligne\]](#)

--enable-dba=shared

- PHP 3: Option non disponible en PHP 3
PHP 4: Compile DBA comme librairie partagée

[4.3.1.3 install.configure.enable-dbase](#)

[\[Notes en ligne\]](#)

--enable-dbase

- PHP 3: Option non disponible; utilisez [4.3.1.4 install.configure.with-dbase](#).
PHP 4: Active la librairie dbase. Par de librairie externe nécessaire.

[4.3.1.4 install.configure.with-dbase](#)

[\[Notes en ligne\]](#)

--with-dbase

- PHP 3: Inclus la librairie dbase. Par de librairie externe nécessaire.
PHP 4: Option non disponible; utilisez [4.3.1.3 install.configure.enable-dbase](#).

[4.3.1.5 install.configure.with-db2](#)

[\[Notes en ligne\]](#)

--with-db2[=DIR]

- PHP 3, PHP 4: Ajoute le support de Berkeley DB2

[4.3.1.6 install.configure.with-db3](#)

[\[Notes en ligne\]](#)

--with-db3[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support Berkeley DB3

[4.3.1.7 install.configure.with-dbm](#)

[\[Notes en ligne\]](#)

--with-dbm[=DIR]

- PHP 3, PHP 4: Inclus le support DBM

[4.3.1.8 install.configure.with-dbmaker](#)

[\[Notes en ligne\]](#)

--with-dbmaker[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support DBMaker. DIR est le dossier de l'installation de DBMaker (par défaut, c'est le chemin jusqu'à la dernière installation de DBMaker, ou /home/dbmaker/3.6).

[4.3.1.9 install.configure.with-empress](#)

[\[Notes en ligne\]](#)

--with-empress[=DIR]

- PHP 3, PHP 4: Inclus le support Empress. DIR est le dossier d'installation d'Empress (par défaut, \$EMPRESSPATH).

[4.3.1.10 install.configure.enable-filepro](#)

[\[Notes en ligne\]](#)

--enable-filepro

- PHP 3: Option non disponible; utilisez [4.3.1.11 install.configure.with-filepro](#).
PHP 4: Active le support de filePro (lecture seule). Par de librairie externe nécessaire.

[4.3.1.11 install.configure.with-filepro](#)

[\[Notes en ligne\]](#)

--with-filepro

- PHP 3: Active le support de filePro (lecture seule). Par de librairie externe nécessaire.
PHP 4: Option non disponible; utilisez [4.3.1.10 install.configure.enable-filepro](#).

[4.3.1.12 install.configure.with-gdbm](#)

[\[Notes en ligne\]](#)

--with-gdbm[=DIR]

- PHP 3, PHP 4: Inclus le support GDBM

[4.3.1.13 install.configure.with-hyperwave](#)

[\[Notes en ligne\]](#)

--with-hyperwave

- PHP 3, PHP 4: Inclus le support Hyperwave

[4.3.1.14 install.configure.with-ibm-db2](#)

[\[Notes en ligne\]](#)

--with-ibm-db2[=DIR]

- PHP 3, PHP 4: Inclus le support IBM DB2. DIR est le chemin jusqu'au dossier d'installation de DB2 (par défaut ``/home/db2inst1/sqlllib'`).

[Site IBM DB2](#)

[4.3.1.15 install.configure.with-informix](#)

[\[Notes en ligne\]](#)

--with-informix[=DIR]

- PHP 3, PHP 4: Inclus le support Informix. DIR est le dossier d'installation de Informix (pas de valeur par défaut).

[4.3.1.16 install.configure.with-ingres](#)

[\[Notes en ligne\]](#)

--with-ingres[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support Ingres II support. DIR est le dossier d'installation de Ingres (par défaut `/II/ingres`)

[4.3.1.17 install.configure.with-interbase](#)

[\[Notes en ligne\]](#)

--with-interbase[=DIR]

- PHP 3, PHP 4: Inclus le support InterBase. DIR est le dossier d'installation de InterBase (par défaut ``/usr/interbase'`).

[10.29 Fonctions InterBase](#)

[Site Interbase](#)

[4.3.1.18 install.configure.with-ldap](#)

[\[Notes en ligne\]](#)

--with-ldap[=DIR]

- PHP 3: Inclus le support LDAP. DIR est le dossier d'installation de LDAP (par défaut ``/usr'` et ``/usr/local'`).
PHP 4: Inclus le support LDAP. DIR est le dossier d'installation de LDAP.
Cette option fournit un accès aux serveurs **LDAP** (Lightweight Directory Access Protocol support).
Le paramètre est le dossier d'installation de LDAP (par défaut, ``/usr/local/ldap'`).
Pour plus de détails sur LDAP allez à [RFC1777](#) et [RFC1778](#).

[4.3.1.19 install.configure.with-msql](#)

[\[Notes en ligne\]](#)

--with-msql[=DIR]

- PHP 3, PHP 4: Active le support mSQL. Le paramètre est le dossier d'installation de mSQL (par défaut, ``/usr/local/Hughes'`, le dossier par défaut de l'installation mSQL 2.0). configure automatiquement détecte quelle version de mSQL vous utilisez et gère le point avec les deux versions 1.0 et 2.0, mais si vous compilez PHP avec mSQL 1.0, vous ne pourrez accéder qu'aux bases mSQL 1.0, et vice-versa.
Voir aussi les directives de [7.1.7 Directives de configuration mSQL](#) du fichier [7.1 Le fichier de configuration](#).
[site mSQL](#)

[4.3.1.20 install.configure.with-mysql](#)

[\[Notes en ligne\]](#)

--with-mysql[=DIR]

- PHP 3: Inclus le support MySQL. DIR est le dossier d'installation de MySQL (par défaut, il recherche dans les lieux standards d'installation de MySQL).
PHP 4: Inclus le support MySQL. DIR est le dossier d'installation de MySQL. Si il n'est pas spécifié, la librairie MySQL sera utilisée (option par défaut).
Voir aussi les directives de [7.1.6 MySQL Configuration Directives](#) du fichier de [7.1 Le fichier de configuration](#).
[Site de MySQL](#)

[4.3.1.21 install.configure.with-ndbm](#)

[\[Notes en ligne\]](#)

--with-ndbm[=DIR]

- PHP 3, PHP 4: Inclus le support NDBM

[4.3.1.22 install.configure.with-oci8](#)

[\[Notes en ligne\]](#)

--with-oci8[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support Oracle-oci8. DIR est le dossier d'installation de Oracle (par défaut ORACLE_HOME).

[4.3.1.23 install.configure.with-oracle](#)

[\[Notes en ligne\]](#)

--with-oracle[=DIR]

- PHP 3: Inclus le support Oracle . DIR est le dossier d'installation de Oracle (par défaut \$ORACLE_HOME).
PHP 4: Inclus le support Oracle-oci7. DIR est le dossier d'installation de Oracle (par défaut

ORACLE_HOME).

Inclus le support Oracle. Cette option a été testée et devrait fonctionner au moins avec les versions d'Oracle de 7.0 à 7.3. Le paramètre DIR est *ORACLE_HOME*. Vous n'avez pas à spécifier ce paramètre, si votre environnement Oracle a été installé.

[Site Oracle](#)

[4.3.1.24 install.configure.with-pgsql](#)

[\[Notes en ligne\]](#)

--with-pgsql[=DIR]

- PHP 3: Inclus le support PostgreSQL. DIR est le dossier d'installation de PostgreSQL (par défaut ``usr/local/pgsql'`).
 - PHP 4: Inclus le support PostgreSQL. DIR est le dossier d'installation de PostgreSQL (par défaut ``usr/local/pgsql'`). Utilisez "shared" pour compiler PostgreSQL en librairie dynamique, ou "shared,DIR" pour compiler PostgreSQL en librairie dynamique tout en spécifiant DIR.
- Voir aussi les directives de configurations de [7.1.8 Directives de configuration Postgres](#) dans le fichier de [7.1 Le fichier de configuration](#).

[Site PostgreSQL](#)

[4.3.1.25 install.configure.with-solid](#)

[\[Notes en ligne\]](#)

--with-solid[=DIR]

- PHP 3, PHP 4: Inclus le support Solid. DIR est le dossier d'installation de Solid (par défaut `/usr/local/solid`).

[Site Solid](#)

[4.3.1.26 install.configure.with-sybase-ct](#)

[\[Notes en ligne\]](#)

--with-sybase-ct[=DIR]

- PHP 3, PHP 4: Inclus le support Sybase-CT. DIR est le dossier d'installation de Sybase (par défaut `/home/sybase`).
- Voir aussi les directives de configuration de [7.1.10 Sybase-CT Configuration Directives](#) dans le fichier de [7.1 Le fichier de configuration](#).

[4.3.1.27 install.configure.with-sybase](#)

[\[Notes en ligne\]](#)

--with-sybase[=DIR]

- PHP 3, PHP 4: Inclus le support Sybase-DB. DIR est le dossier d'installation de Sybase (par défaut ``home/sybase'`).
- Voir aussi les directives de configuration de [7.1.9 Directives de configuration Sybase](#) dans le dossier de [7.1 Le fichier de configuration](#).

[site Sybase](#)

[4.3.1.28 install.configure.with-openlink](#)

[\[Notes en ligne\]](#)

--with-openlink[=DIR]

- PHP 3, PHP 4: Inclus le support OpenLink ODBC . DIR est le dossier d'installation de OpenLink (par défaut /usr/local/openlink).

[Site OpenLink Software](#)

[4.3.1.29 install.configure.with-iodbc](#)

[\[Notes en ligne\]](#)

--with-iodbc[=DIR]

- PHP 3, PHP 4: Inclus le support iODBC. DIR est le dossier d'installation de iODBC (par défaut ``/usr/local'`).

Cette option a été développée pour le gestionnaire de pilote iODBC, qui était librement redistribuable, et fonctionnait sous la plus part des UNIX.

[Site FreeODBC](#) or [Site iODBC](#)

[4.3.1.30 install.configure.with-custom-odbc](#)

[\[Notes en ligne\]](#)

--with-custom-odbc[=DIR]

- PHP 3, PHP 4: Inclus le support pour une librairie arbitraire ODBC. Le paramètre est le dossier d'installation et par défaut, c'est ``/usr/local'`. Cette option implique que vous avez défini la variable CUSTOM_ODBC_LIBS, lorsque vous exécutez le script de configuration. Vous devez aussi fournir un entête odbc.h valide, dans le chemin d'inclusion. Si vous n'en avez pas, créez le, et incluez y vos entêtes spécifiques. Ces entêtes peuvent aussi utiliser des définitions supplémentaires, surtout si ils sont multi-plateformes. Définissez les dans CFLAGS.

Par exemple, vous pouvez utiliser Sybase SQL Anywhere sous QNX comme suit :

```
CFLAGS=-DODBC_QNX LDFLAGS=-linux CUSTOM_ODBC_LIBS="-ldblib -lodbc"
./configure --with-custom-odbc=/usr/lib/sqlany50
```

[4.3.1.31 install.configure.disable-unified-odbc](#)

[\[Notes en ligne\]](#)

--disable-unified-odbc

- PHP 3: Inactive le support ODBC unifié. Uniquement utile si iODBC, Adabas, Solid, Velocis ou une interface ODBC personnalisée est activée.

PHP 4: Option non disponible en PHP 4

Le module ODBC unifié, qui est une interface répandue vers toutes les bases de données ODBC, comme par exemple Solid, IBM DB2 et Adabas D. Elle fonctionne aussi pour les librairie ODBC normales. Elle a été testée avec iODBC, Solid, Adabas D, IBM DB2 et Sybase SQL Anywhere. Requiert qu'un (et seulement un) de ces module ait été activé, ou que le module Velocis soit activé, ou encore que la librairie ODBC personnalisée soit chargée. Cette option ne sert que si l'une de ces options est utilisée : [4.3.1.29 install.configure.with-iodbc](#), [4.3.1.25 install.configure.with-solid](#), [4.3.1.14 install.configure.with-ibm-db2](#), [4.3.1.1 install.configure.with-adabas](#), [4.3.1.33](#)

[install.configure.with-velocis](#), ou [4.3.1.30 install.configure.with-custom-odbc](#).

Voir aussi les directives de configuration de [7.1.14 Directives de configuration du driver ODBC unifié](#) dans le fichier de [7.1 Le fichier de configuration](#).

[4.3.1.32 install.configure.with-unixODBC](#)

[\[Notes en ligne\]](#)

--with-unixODBC[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support unixODBC. DIR est le dossier d'installation de unixODBC (par défaut /usr/local).

[4.3.1.33 install.configure.with-velocis](#)

[\[Notes en ligne\]](#)

--with-velocis[=DIR]

- PHP 3, PHP 4: Inclus le support Velocis. DIR est le dossier d'installation de Velocis (par défaut /usr/local/velocis).
[Site Velocis](#)

[4.3.2 Ecommerce](#)

[\[Notes en ligne\]](#)

[4.3.2.1 install.configure.with-ccvs](#)

[\[Notes en ligne\]](#)

--with-ccvs[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support CCVS de PHP 4. Passez le dossier d'installation de CCVS comme valeur de DIR.

[4.3.2.2 install.configure.with-mck](#)

[\[Notes en ligne\]](#)

--with-mck[=DIR]

- PHP 3: Inclus le support Cybercash MCK. DIR est le dossier d'installation de cybercash mck (par défaut ``/usr/src/mck-3.2.0.3-linux'`). Pour plus d'informations, ``extra/cyberlib'`.
PHP 4: Option non disponible; utilisez [4.3.2.3 install.configure.with-cybercash](#).

[4.3.2.3 install.configure.with-cybercash](#)

[\[Notes en ligne\]](#)

--with-cybercash[=DIR]

- PHP 3: Option non disponible; utilisez plutôt [4.3.2.2 install.configure.with-mck](#).
PHP 4: Inclus le support CyberCash. DIR est le dossier d'installation de CyberCash MCK.

[4.3.2.4 install.configure.with-pfpro](#)

[\[Notes en ligne\]](#)

--with-pfpro[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus de support Verisign Payflow Pro.

[4.3.3 Graphisme](#)

[\[Notes en ligne\]](#)

[4.3.3.1 install.configure.enable-freetype-4bit-antialias-hack](#)

[\[Notes en ligne\]](#)

--enable-freetype-4bit-antialias-hack

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support de FreeType2 (expérimental).

[4.3.3.2 install.configure.with-gd](#)

[\[Notes en ligne\]](#)

--with-gd[=DIR]

- PHP 3: Inclus le support GD (DIR est le dossier d'installation de GD).
PHP 4: Inclus le support GD (DIR est le dossier d'installation de GD). Utilisez "shared" pour compiler GD en librairie dynamique, ou "shared,DIR" pour compiler GD en librairie dynamique tout en spécifiant DIR.

[4.3.3.3 install.configure.without-gd](#)

[\[Notes en ligne\]](#)

--without-gd

- PHP 3, PHP 4: Inactive le support GD.

[4.3.3.4 install.configure.with-imagick](#)

[\[Notes en ligne\]](#)

--with-imagick[=DIR]

- PHP 3: Inclus le support ImageMagick. DIR est le dossier d'installation (par défaut, PHP essaiera de se débrouiller).[expérimental]
PHP 4: Option non disponible en PHP 4

[4.3.3.5 install.configure.with-jpeg-dir](#)

[\[Notes en ligne\]](#)

--with-jpeg-dir[=DIR]

- PHP 3: Dossier jpeg pour pdflib 2.0
PHP 4: Dossier jpeg pour pdflib 3.x

[4.3.3.6 install.configure.with-png-dir](#)

[\[Notes en ligne\]](#)

--with-png-dir[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Dossier png pour pdflib 3.x

[4.3.3.7 install.configure.enable-t1lib](#)

[\[Notes en ligne\]](#)

--enable-t1lib

- PHP 3: Active le support t1lib.
PHP 4: Option non disponible; utilisez plutôt [4.3.3.8 install.configure.with-t1lib](#).

[4.3.3.8 install.configure.with-t1lib](#)

[\[Notes en ligne\]](#)

--with-t1lib[=DIR]

- PHP 3: Option non disponible; utilisez plutôt [4.3.3.7 install.configure.enable-t1lib](#).
PHP 4: Inclus le support t1lib.

[4.3.3.9 install.configure.with-tiff-dir](#)

[\[Notes en ligne\]](#)

--with-tiff-dir[=DIR]

- PHP 3: Dossier tiff pour pdflib 2.0
PHP 4: Dossier tiff pour pdflib 3.x

[4.3.3.10 install.configure.with-ttf](#)

[\[Notes en ligne\]](#)

--with-ttf[=DIR]

- PHP 3, PHP 4: Inclus le support ttf

[4.3.3.11 install.configure.with-xpm-dir](#)

[\[Notes en ligne\]](#)

--with-xpm-dir[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Dossier xpm pour gd-1.8+

4.3.4 Divers

[\[Notes en ligne\]](#)

Cette section est en cours de tri.

4.3.4.1 install.configure.disable-bcmath

[\[Notes en ligne\]](#)

--disable-bcmath

- PHP 3: Inactive la librairie de nombre BC (taille arbitraire). Ces fonctions vous permettent de manipuler des nombres hors des limites habituelles des entiers et des nombres à virgules flottantes. Voir aussi [10.4 Fonctions mathématiques sur des nombres de taille arbitraire](#).
PHP 4: Option non disponible; bcmath n'est pas compilé par défaut. Utilisez [4.3.4.9 install.configure.enable-bcmath](#).

4.3.4.2 install.configure.disable-display-source

[\[Notes en ligne\]](#)

--disable-display-source

- PHP 3: Compilation sans affichage du source.
PHP 4: Option non disponible en PHP 4

4.3.4.3 install.configure.disable-libtool-lock

[\[Notes en ligne\]](#)

--disable-libtool-lock

- PHP 3: Option non disponible en PHP 3
PHP 4: Empêche le verrouillage (peut bloquer certaines compilations parallèle).

4.3.4.4 install.configure.disable-pear

[\[Notes en ligne\]](#)

--disable-pear

- PHP 3: Option non disponible en PHP 3
PHP 4: N'installe pas PEAR

4.3.4.5 install.configure.disable-pic

[\[Notes en ligne\]](#)

--disable-pic

- PHP 3: Option non disponible en PHP 3
PHP 4: Inactive PIC pour les objets partagés.

[4.3.4.6 install.configure.disable-posix](#)

[\[Notes en ligne\]](#)

--disable-posix

- PHP 3: Option non disponible en PHP 3; utilisez [4.3.4.61 install.configure.without-posix](#).
PHP 4: Désactive les fonctions POSIX.

[4.3.4.7 install.configure.disable-rpath](#)

[\[Notes en ligne\]](#)

--disable-rpath

- PHP 3: Option non disponible en PHP 3
PHP 4: Inactive le passage automatique de dossier de recherche supplémentaires.

[4.3.4.8 install.configure.disable-session](#)

[\[Notes en ligne\]](#)

--disable-session

- PHP 3: Option non disponible en PHP 3
PHP 4: Inactive les sessions

[4.3.4.9 install.configure.enable-bcmath](#)

[\[Notes en ligne\]](#)

--enable-bcmath

- PHP 3: Option non disponible en PHP 3; bcmath est compilé par Utilisez [4.3.4.1 install.configure.disable-bcmath](#) pour le désactiver.
PHP 4: Compile PHP avec les fonctions mathématiques BC. Lisez le fichier README-BCMATH pour connaître les instructions d'installation de ce module. Ces fonctions vous permettent de travailler avec des nombres de tailles illimitées. Reportez vous aux fonctions [10.4 Fonctions mathématiques sur des nombres de taille arbitraire](#) pour plus d'informations.

[4.3.4.10 install.configure.enable-c9x-inline](#)

[\[Notes en ligne\]](#)

--enable-c9x-inline

- PHP 3: Option non disponible en PHP 3
PHP 4: Active la sémantique C9x-inline.

[4.3.4.11 install.configure.enable-calendar](#)

[\[Notes en ligne\]](#)

--enable-calendar

- PHP 3: Option non disponible en PHP 3
PHP 4: Active le support des conversions calendaires

[4.3.4.12 install.configure.enable-debug](#)

[\[Notes en ligne\]](#)

--enable-debug

- PHP 3, PHP 4: Ajoute le débogage des symboles.

[4.3.4.13 install.configure.enable-debugger](#)

[\[Notes en ligne\]](#)

--enable-debugger

- PHP 3: Ajoute les fonctions de débogage distants.
PHP 4: Option non disponible en PHP 4

[4.3.4.14 install.configure.enable-discard-path](#)

[\[Notes en ligne\]](#)

--enable-discard-path

- PHP 3, PHP 4: Si cette option est activée, l'exécutable PHP CGI peut être placé sûrement hors de l'arborescence web, et personne ne pourra contourner la sécurité .htaccess.

[4.3.4.15 install.configure.enable-dmalloc](#)

[\[Notes en ligne\]](#)

--enable-dmalloc

- PHP 3, PHP 4: Active dmalloc

[4.3.4.16 install.configure.enable-exif](#)

[\[Notes en ligne\]](#)

--enable-exif

- PHP 3: Option non disponible en PHP 3
PHP 4: Active le support exif

[4.3.4.17 install.configure.enable-experimental-zts](#)

[\[Notes en ligne\]](#)

--enable-experimental-zts

- PHP 3: Option non disponible en PHP 3
PHP 4: Cette option risque de détruire votre compilation

[4.3.4.18 install.configure.enable-fast-install](#)

[\[Notes en ligne\]](#)

--enable-fast-install[=PKGS]

- PHP 3: Option non disponible en PHP 3
PHP 4: Optimise la vitesse d'exécution [par défaut=yes]

[4.3.4.19 install.configure.enable-force-cgi-redirect](#)

[\[Notes en ligne\]](#)

--enable-force-cgi-redirect

- PHP 3, PHP 4: Active la vérification des redirections internes au serveur. Il est recommandé d'utiliser cette option si vous utilisez la version CGI avec Apache.

[4.3.4.20 install.configure.enable-inline-optimization](#)

[\[Notes en ligne\]](#)

--enable-inline-optimization

- PHP 3: Option non disponible en PHP 3
PHP 4: Si vous avez beaucoup de mémoire libre, et que vous utilisez gcc, vous pouvez essayer cela.

[4.3.4.21 install.configure.enable-libgcc](#)

[\[Notes en ligne\]](#)

--enable-libgcc

- PHP 3: Option non disponible en PHP 3
PHP 4: Active explicitement le linkage avec libgcc

[4.3.4.22 install.configure.enable-maintainer-mode](#)

[\[Notes en ligne\]](#)

--enable-maintainer-mode

- PHP 3, PHP 4: Active des règles de compilation rament utile (et même parfois confuse) à l'installateur occasionnel.

[4.3.4.23 install.configure.enable-memory-limit](#)

[\[Notes en ligne\]](#)

--enable-memory-limit

- PHP 3, PHP 4: Compile avec limitation de consommation de mémoire.

[4.3.4.24 install.configure.enable-safe-mode](#)

[\[Notes en ligne\]](#)

--enable-safe-mode

- PHP 3, PHP 4: Active le safe mode par défaut.

[4.3.4.25 install.configure.enable-satellite](#)

[\[Notes en ligne\]](#)

--enable-satellite

- PHP 3: Option non disponible en PHP 3
PHP 4: Active le support CORBA via Satellite (Requiert ORBit)

[4.3.4.26 install.configure.enable-shared](#)

[\[Notes en ligne\]](#)

--enable-shared[=PKGS]

- PHP 3: Option non disponible en PHP 3
PHP 4: Compile les bibliothèques partagées [par défaut=yes]

[4.3.4.27 install.configure.enable-sigchild](#)

[\[Notes en ligne\]](#)

--enable-sigchild

- PHP 3, PHP 4: Active le gestionnaire SIGCHLD interne à PHP.

[4.3.4.28 install.configure.enable-static](#)

[\[Notes en ligne\]](#)

--enable-static[=PKGS]

- PHP 3: Option non disponible en PHP 3
PHP 4: Compile les bibliothèques statiquement [par défaut=yes]

[4.3.4.29 install.configure.enable-sysvsem](#)

[\[Notes en ligne\]](#)

--enable-sysvsem

- PHP 3, PHP 4: Active le support des sémaphores System V.

[4.3.4.30 install.configure.enable-sysvshm](#)

[\[Notes en ligne\]](#)

--enable-sysvshm

- PHP 3, PHP 4: Active le système System V de mémoire partagée

[4.3.4.31 install.configure.enable-trans-sid](#)

[\[Notes en ligne\]](#)

--enable-trans-sid

- PHP 3: Option non disponible en PHP 3

PHP 4: Active la propagation transparente d'identification de session

[4.3.4.32 install.configure.with-cdb](#)

[\[Notes en ligne\]](#)

--with-cdb[=DIR]

- PHP 3, PHP 4: Inclus le support cdb

[4.3.4.33 install.configure.with-config-file-path](#)

[\[Notes en ligne\]](#)

--with-config-file-path=PATH

- PHP 3: Indique le chemin dans lequel il faut chercher le fichier php3.ini (par défaut, ``/usr/local/lib'`).
- PHP 4: Indique le chemin dans lequel il faut chercher le fichier php.ini (par défaut, ``/usr/local/lib'`).

[4.3.4.34 install.configure.with-cpdflib](#)

[\[Notes en ligne\]](#)

--with-cpdflib[=DIR]

- PHP 3: Inclus le support cpdflib. DIR est le dossier d'installation de ClibPDF (par défaut, `/usr/local`).
- PHP 4: Inclus le support cpdflib (requiert cpdflib >= 2). DIR est le dossier d'installation de ClibPDF (par défaut, `/usr/local`).

[4.3.4.35 install.configure.with-esoob](#)

[\[Notes en ligne\]](#)

--with-esoob[=DIR]

- PHP 3: Option non disponible en PHP 3
- PHP 4: Inclus le support Easysoft OOB. DIR est le dossier d'installation de OOB (par défaut `/usr/local/easysoft/oob/client`).

[4.3.4.36 install.configure.with-exec-dir](#)

[\[Notes en ligne\]](#)

--with-exec-dir[=DIR]

- PHP 3, PHP 4: Autorise uniquement les exécutables dans le dossier DIR lorsque le safe mode est activé (par défaut, `/usr/local/php/bin`).

[4.3.4.37 install.configure.with-fdftk](#)

[\[Notes en ligne\]](#)

--with-fdftk[=DIR]

- PHP 3, PHP 4: Inclus le support fdftk. DIR est le dossier d'installation de fdftk (par défaut, `/usr/local`).

[4.3.4.38 install.configure.with-gnu-ld](#)

[\[Notes en ligne\]](#)

--with-gnu-ld

- PHP 3: Option non disponible en PHP 3
PHP 4: Support que le compilateur C utilise GNU ld [par défaut =no]

[4.3.4.39 install.configure.with-icap](#)

[\[Notes en ligne\]](#)

--with-icap[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support ICAP.

[4.3.4.40 install.configure.with-imap](#)

[\[Notes en ligne\]](#)

--with-imap[=DIR]

- PHP 3, PHP 4: Inclus le support IMAP. DIR est le dossier d'installation de IMAP (celui qui contient les include et c-client.a).

[4.3.4.41 install.configure.with-imspp](#)

[\[Notes en ligne\]](#)

--with-imspp[=DIR]

- PHP 3: Inclus le support IMSP (DIR est le dossier d'installation de IMSP (celui qui contient les include et c-client.a).
PHP 4: Option non disponible en PHP 4

[4.3.4.42 install.configure.with-java](#)

[\[Notes en ligne\]](#)

--with-java[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support Java. DIR est le dossier d'installation du JDK. Cette extension ne peut être compilé que comme une librairie partagée.

[4.3.4.43 install.configure.with-kerberos](#)

[\[Notes en ligne\]](#)

--with-kerberos[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support Kerberos avec IMAP.

[4.3.4.44 install.configure.with-mcal](#)

[\[Notes en ligne\]](#)

--with-mcal[=DIR]

- PHP 3, PHP 4: Inclus le support MCAL.

[4.3.4.45 install.configure.with-mcrypt](#)

[\[Notes en ligne\]](#)

--with-mcrypt[=DIR]

- PHP 3, PHP 4: Inclus le support mcrypt. DIR est le dossier d'installation de mcrypt.

[4.3.4.46 install.configure.with-mhash](#)

[\[Notes en ligne\]](#)

--with-mhash[=DIR]

- PHP 3, PHP 4: Inclus le support mhash. DIR est le dossier d'installation de mhash.

[4.3.4.47 install.configure.with-mm](#)

[\[Notes en ligne\]](#)

--with-mm[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support mm pour le stockage des sessions

[4.3.4.48 install.configure.with-mod_charset](#)

[\[Notes en ligne\]](#)

--with-mod_charset

- PHP 3, PHP 4: Active les tables de transfert pour mod_charset (Rus Apache).

[4.3.4.49 install.configure.with-pdflib](#)

[\[Notes en ligne\]](#)

--with-pdflib[=DIR]

- PHP 3: Inclus le support pdflib (testée avec 0.6 et 2.0). DIR est le dossier d'installation de pdflib (par défaut, ``/usr/local'`).
- PHP 4: Inclus le support pdflib 3.x. DIR est le dossier d'installation de pdflib (par défaut, ``/usr/local'`).

[4.3.4.50 install.configure.with-readline](#)

[\[Notes en ligne\]](#)

--with-readline[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support readline. DIR est le dossier d'installation de readline.

[4.3.4.51 install.configure.with-regex](#)

[\[Notes en ligne\]](#)

--with-regex=TYPE

- PHP 3: Option non disponible en PHP 3
PHP 4: Librairie de regex : system, apache, php

[4.3.4.52 install.configure.with-servlet](#)

[\[Notes en ligne\]](#)

--with-servlet[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support des servlets . DIR est le dossier d'installation du JSDK. Ces SAPI requiert la compilation de l'extension Java comme librairie partagée.

[4.3.4.53 install.configure.with-swf](#)

[\[Notes en ligne\]](#)

--with-swf[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support swf.

[4.3.4.54 install.configure.with-system-regex](#)

[\[Notes en ligne\]](#)

--with-system-regex

- PHP 3: N'utilise pas la librairie de regex fournie.
PHP 4: (obsolète) Utilise la librairie de regex.

[4.3.4.55 install.configure.with-tsrm-pth](#)

[\[Notes en ligne\]](#)

--with-tsrm-pth[=pth-config]

- PHP 3: Option non disponible en PHP 3
PHP 4: Utilise GNU Pth.

[4.3.4.56 install.configure.with-tsrm-pthreads](#)

[\[Notes en ligne\]](#)

--with-tsrm-pthreads

- PHP 3: Option non disponible en PHP 3
PHP 4: Utilise les threads POSIX (par défaut)

[4.3.4.57 install.configure.with-x](#)

[\[Notes en ligne\]](#)

--with-x

- PHP 3: Utilise le système X Window
PHP 4: Option non disponible en PHP 4

[4.3.4.58 install.configure.with-zlib-dir](#)

[\[Notes en ligne\]](#)

--with-zlib-dir[=DIR]

- PHP 3: Dossier zlib pour pdflib 2.0 ou Inclus le support zlib
PHP 4: Dossier zlib pour pdflib 3.x ou Inclus le support zlib

[4.3.4.59 install.configure.with-zlib](#)

[\[Notes en ligne\]](#)

--with-zlib[=DIR]

- PHP 3, PHP 4: Inclus le support zlib (requiert zlib >= 1.0.9). DIR est le dossier d'installation de zlib (par défaut, /usr).

[4.3.4.60 install.configure.without-pcre-regex](#)

[\[Notes en ligne\]](#)

--without-pcre-regex

- PHP 3: Exclut les expressions régulières compatibles Perl.
PHP 4: Exclut les expressions régulières compatibles Perl. Use *--with-pcre-regex=DIR* pour spécifier le dossier DIR qui contient les dossier d'include et de libraries, si vous n'utilisez pas la librairie fournie.

[4.3.4.61 install.configure.without-posix](#)

[\[Notes en ligne\]](#)

--without-posix

- PHP 3: Exclut les fonctions POSIX
PHP 4: Option non disponible en PHP 4; utilisez [4.3.4.6 install.configure.disable-posix](#).

[4.3.5 Réseau](#)

[\[Notes en ligne\]](#)

[4.3.5.1 install.configure.with-curl](#)

[\[Notes en ligne\]](#)

--with-curl[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support CURL

[4.3.5.2 install.configure.enable-ftp](#)

[\[Notes en ligne\]](#)

--enable-ftp

- PHP 3: Option non disponible; utilisez plutôt [4.3.5.3 install.configure.with-ftp](#).
PHP 4: Enable FTP support

[4.3.5.3 install.configure.with-ftp](#)

[\[Notes en ligne\]](#)

--with-ftp

- PHP 3: Inclus le support FTP.
PHP 4: Option non disponible; utilisez plutôt [4.3.5.2 install.configure.enable-ftp](#) instead

[4.3.5.4 install.configure.disable-url-fopen-wrapper](#)

[\[Notes en ligne\]](#)

--disable-url-fopen-wrapper

- PHP 3, PHP 4: Inactive la version améliorée de fopen, qui permet l'accès aux fichiers par HTTP ou FTP.

[4.3.5.5 install.configure.with-mod-dav](#)

[\[Notes en ligne\]](#)

--with-mod-dav=DIR

- PHP 3, PHP 4: Inclus le support DAV grâce à mod_dav, DIR est le dossier d'installation de mod_dav's (version Apache module seulement!)

[4.3.5.6 install.configure.with-openssl](#)

[\[Notes en ligne\]](#)

--with-openssl[=DIR]

- PHP 3, PHP 4: Inclus le support OpenSSL de SNMP.

[4.3.5.7 install.configure.with-snmpp](#)

[\[Notes en ligne\]](#)

--with-snmpp[=DIR]

- PHP 3, PHP 4: Inclus le support SNMP. DIR est le dossier d'installation de SNMP (par défaut, PHP va chercher dans les un certains nombre d'endroits classiques d'installation de snmp). Utilisez "shared" pour compiler SNMP en librairie dynamique, ou "shared,DIR" pour compiler SNMP en librairie dynamique tout en spécifiant DIR.

[4.3.5.8 install.configure.enable-ucd-snmp-hack](#)

[\[Notes en ligne\]](#)

--enable-ucd-snmp-hack

- PHP 3, PHP 4: Active le hack UCD SNMP

[4.3.5.9 install.configure.enable-sockets](#)

[\[Notes en ligne\]](#)

--enable-sockets

- PHP 3: Option non disponible en PHP 3
PHP 4: Active le support des sockets

[4.3.5.10 install.configure.with-yaz](#)

[\[Notes en ligne\]](#)

--with-yaz[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support YAZ (ANSI/NISO Z39.50). DIR est le dossier d'installation de YAZ.

[4.3.5.11 install.configure.enable-yp](#)

[\[Notes en ligne\]](#)

--enable-yp

- PHP 3: Option non disponible; utilisez plutôt [4.3.5.12 install.configure.with-yp](#).
PHP 4: Inclus le support YP

[4.3.5.12 install.configure.with-yp](#)

[\[Notes en ligne\]](#)

--with-yp

- PHP 3: Inclus le support YP
PHP 4: Option non disponible; utilisez plutôt [4.3.5.11 install.configure.enable-yp](#).

[4.3.6 Configuration de PHP](#)

[\[Notes en ligne\]](#)

[4.3.6.1 install.configure.enable-magic-quotes](#)

[\[Notes en ligne\]](#)

--enable-magic-quotes

- PHP 3, PHP 4: Active les magic quotes par défaut.

[4.3.6.2 install.configure.disable-short-tags](#)

[\[Notes en ligne\]](#)

--disable-short-tags

- PHP 3, PHP 4: Désactive les balises courtes (<?).

[4.3.6.3 install.configure.enable-track-vars](#)

[\[Notes en ligne\]](#)

--enable-track-vars

- PHP 3: Active la lecture des variables GET/POST/Cookie par défaut.
PHP 4: Option non disponible en PHP 4; sous PHP 4.0.2, track_vars est toujours à on.

[4.3.7 Serveur](#)

[\[Notes en ligne\]](#)

[4.3.7.1 install.configure.with-aolserver-src](#)

[\[Notes en ligne\]](#)

--with-aolserver-src=DIR

- PHP 3: Option non disponible en PHP 3
PHP 4: Spécifie le chemin jusqu'à la distribution du serveur AOLserver

[4.3.7.2 install.configure.with-aolserver](#)

[\[Notes en ligne\]](#)

--with-aolserver=DIR

- PHP 3: Option non disponible en PHP 3
PHP 4: Spécifie le chemin jusqu'au serveur AOLserver

[4.3.7.3 install.configure.with-apache](#)

[\[Notes en ligne\]](#)

--with-apache[=DIR]

- PHP 3, PHP 4: Compile un module Apache. DIR est le dossier d'installation d'Apache (par défaut, /usr/local/etc/httpd).

[4.3.7.4 install.configure.with-apxs](#)

[\[Notes en ligne\]](#)

--with-apxs[=FILE]

- PHP 3, PHP 4: Compile un module partagé Apache. FILE est un chemin optionnel jusqu'aux outils Apache apxs (par défauts, apxs).

[4.3.7.5 install.configure.enable-versioning](#)

[\[Notes en ligne\]](#)

--enable-versioning

- PHP 3: Profite du versioning et du scoping de Solaris 2.x et Linux
PHP 4: Export uniquement les symboles requis. Voir INSTALL pour plus d'informations.

[4.3.7.6 install.configure.with-fhttpd](#)

[\[Notes en ligne\]](#)

--with-fhttpd[=DIR]

- PHP 3, PHP 4: Compile un module fhttpd. DIR est le dossier d'installation de fhttpd (par défaut, /usr/local/src/fhttpd).

[4.3.7.7 install.configure.with-nsapi](#)

[\[Notes en ligne\]](#)

--with-nsapi=DIR

- PHP 3: Option non disponible en PHP 3
PHP 4: Spécifie le chemin jusqu'à Netscape

[4.3.7.8 install.configure.with-phttpd](#)

[\[Notes en ligne\]](#)

--with-phttpd=DIR

- PHP 3: Option non disponible en PHP 3
PHP 4:

[4.3.7.9 install.configure.with-pi3web](#)

[\[Notes en ligne\]](#)

--with-pi3web=DIR

- PHP 3: Option non disponible en PHP 3
PHP 4: Compile PHP comme module pour Pi3Web.

[4.3.7.10 install.configure.with-roxen](#)

[\[Notes en ligne\]](#)

--with-roxen=DIR

- PHP 3: Option non disponible en PHP 3
PHP 4: Compile PHP comme module pour Pike . DIR est le dossier d'installation de Roxen (par défaut, /usr/local/roxen/server).

[4.3.7.11 install.configure.enable-roxen-zts](#)

[\[Notes en ligne\]](#)

--enable-roxen-zts

- PHP 3: Option non disponible en PHP 3
PHP 4: Compile le module Roxen avec Zend Thread Safety.

[4.3.7.12 install.configure.with-thttpd](#)

[\[Notes en ligne\]](#)

--with-thttpd=SRCDIR

- PHP 3: Option non disponible en PHP 3
PHP 4:

[4.3.7.13 install.configure.with-zeus](#)

[\[Notes en ligne\]](#)

--with-zeus=DIR

- PHP 3: Option non disponible en PHP 3
PHP 4: Compile PHP comme module ISAPI pour Zeus.

[4.3.8 Texte et langue](#)

[\[Notes en ligne\]](#)

[4.3.8.1 install.configure.with-aspell](#)

[\[Notes en ligne\]](#)

--with-aspell[=DIR]

- PHP 3, PHP 4: Inclus le support ASPELL.

[4.3.8.2 install.configure.with-gettext](#)

[\[Notes en ligne\]](#)

--with-gettext[=DIR]

- PHP 3, PHP 4: Inclus le support GNU gettext. DIR est le dossier d'installation de gettext (par défaut, /usr/local).

[4.3.8.3 install.configure.with-pspell](#)

[\[Notes en ligne\]](#)

--with-pspell[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support PSPELL.

[4.3.8.4 install.configure.with-recode](#)

[\[Notes en ligne\]](#)

--with-recode[=DIR]

- PHP 3: Include GNU recode support.
PHP 4: Inclus le support recode. DIR est le dossier d'installation de recode.

[4.3.9 XML](#)

[\[Notes en ligne\]](#)

[4.3.9.1 install.configure.with-dom](#)

[\[Notes en ligne\]](#)

--with-dom[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support DOM (requiert libxml >= 2.0). DIR est le dossier d'installation de libxml (par défaut, ``/usr'`).

[4.3.9.2 install.configure.enable-sablot-errors-descriptive](#)

[\[Notes en ligne\]](#)

--enable-sablot-errors-descriptive

- PHP 3: Option non disponible en PHP 3
PHP 4: Active les erreurs descriptives.

[4.3.9.3 install.configure.with-sablot](#)

[\[Notes en ligne\]](#)

--with-sablot[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support Sablotron

[4.3.9.4 install.configure.enable-wddx](#)

[\[Notes en ligne\]](#)

--enable-wddx

- PHP 3: Option non disponible en PHP 3
PHP 4: Active le support WDDX

[4.3.9.5 install.configure.disable-xml](#)

[\[Notes en ligne\]](#)

--disable-xml

- PHP 3: Option non disponible en PHP 3; Les fonctionnalités XML ne sont pas compilées par défaut.

Utilisez [4.3.9.6 install.configure.with-xml](#).

PHP 4: Désactive le support XML qui utilise la librairie fournie expat.

[4.3.9.6 install.configure.with-xml](#)

[\[Notes en ligne\]](#)

`--with-xml`

- PHP 3: Inclus le support XML
- PHP 4: Option non disponible; XML est compilé par défaut. Utilisez [4.3.9.5 install.configure.disable-xml](#) pour le désactiver.

[4.4 Installation sous Windows 95/98/NT](#)

[\[Notes en ligne\]](#)

Ce guide d'installation vous aidera à installer et configurer PHP sur vos serveurs Windows 9x/NT. Ce guide a été compilé par

Ce guide fourni une aide d'installation pour :

- Personal Web Server (Version la plus récente recommandée)
- Internet Information Server 3 ou 4
- Apache 1.3.x
- Omni HTTPd 2.0b1

[4.4.1 Installation](#)

[\[Notes en ligne\]](#)

Les instructions doivent être faites pour toutes les installations avant d'attaquer les instructions spécifiques à chaque serveur.

- Extrayez la distribution dans un dossier de votre choix. "C:\PHP\" est une bonne idée.
- Copiez le fichier, 'php.ini-dist' dans le dossier '%WINDOWS%' et renommez le en 'php.ini'.
'%WINDOWS%' est typiquement
 - ◆ c:\windows pour Windows 95/98
 - ◆ c:\winnt ou c:\winnt40 pour les serveurs NT
- Editez votre fichier 'php.ini':
 - ◆ Vous devez changer votre option 'extension_dir' pour qu'il pointe sur votre dossier d'installation PHP, ou vers l'endroit où vous avez installé vos 'php_*.dll'. ex: c:\php
 - ◆ Si vous utilisez Omni Httpd, sautez l'étape suivante. Modifiez 'doc_root' pour qu'il pointe sur votre racine de serveur web. ex: c:\apache\htdocs ou c:\webroot
 - ◆ Choisissez les modules que vous voulez charger lorsque PHP démarre. Vous pouvez décommenter les lignes 'extension=php_*.dll' pour charger ces modules. Certains modules requièrent que des bibliothèques supplémentaires soient installées sur votre système. La [FAQ](#) PHP a plus d'informations sur ces bibliothèques. Vous pouvez aussi charger dynamiquement ces bibliothèques avec `dl("php_*.dll");`
 - ◆ Sous PWS et IIS, vous pouvez modifier le fichier browscap.ini pour qu'il pointe sur :

'c:\windows\system\inetsrv\browscap.ini' sous Windows 95/98 et
'c:\winnt\system32\inetsrv\browscap.ini' sous NT Plus de détails sur l'utilisation de browscap
sont accessibles sur ce [mirroir](#), sélectionnez le bouton "source" pour le voir en action.

Les DLLs des extensions PHP sont préfixé avec 'php_', pour éviter les confusions entre les extensions PHP et leur librairies.

4.4.2 Windows 95/98/NT et PWS/IIS 3

[\[Notes en ligne\]](#)

La méthode recommandée pour configurer ces serveurs est d'utiliser le fichier INF inclus dans la distribution (php_iis_reg.inf). Vous pouvez éditer ce fichier, pour vous assurer que les extensions et les dossiers d'installation de PHP sont bien ceux de votre configuration. Ou alors, vous pouvez suivre les instructions suivantes :

ATTENTION: Ces instructions requièrent la manipulation du fichier de registry de Windows. Une erreur peut laisser votre système dans un état instable. Nous vous recommandons vivement de sauvegarder ce fichier en lieu sûr. L'équipe de développement ne pourra pas être reconnue responsable d'un quelconque dommage dans votre registry.

- Lancez Regedit.
- Naviguez jusqu'à : HKEY_LOCAL_MACHINE /System /CurrentControlSet /Services /W3Svc /Parameters /ScriptMap.
- Dans le menu edit, sélectionnez : New->String Value.
- Entrez l'extension que vous voulez utiliser pour les scripts PHP. ex: .php
- Double cliquez sur la chaîne, et entrez le chemin jusqu'à php.exe dans le champ "value data". ex: c:\php\php.exe %s %s. Les '%s %s' sont TRES importants, PHP ne fonctionnera pas sans.
- Répétez ces instructions pour toutes les extensions que vous voulez associer aux scripts PHP.
- Naviguez jusqu'à : HKEY_CLASSES_ROOT
- Dans le menu edit, sélectionnez: New->Key.
- Donnez le nom de votre extension à la clé : ex: .php
- Sélectionnez le nom de la nouvelle clé dans le panneau de droite, et double cliquez dans "default value", puis entrez phpfile.
- Répétez ces instructions pour toutes les extensions que vous avez associé aux scripts PHP.
- Créez une autre New->Key sous HKEY_CLASSES_ROOT et nommez la phpfile.
- Sélectionnez la nouvelle clé phpfile et dans le panneau de droite, double cliquez dans "default value" et entrez PHP Script.
- Faites un clic droit dans phpfile et sélectionnez New->Key, appelez-le Shell.
- Faites un clic droit dans Shell et sélectionnez New->Key, appelez-le open.
- Faites un clic droit dans open et sélectionnez New->Key, appelez-le command.
- Sélectionnez la nouvelle clé command et dans le panneau de droite, faites un double clic dans "default value", puis entrez le chemin jusqu'à php.exe. ex: c:\php\php.exe -q %1. (n'oubliez pas le %1).
- Exit Regedit.

Les utilisateurs de PWS et IIS 3 sont prêts à utiliser leur serveur. Avec IIS 3, vous pouvez utiliser un [outil](#) bien pratique de Steven Genusa pour configurer votre carte des scripts.

[4.4.3 Windows NT et IIS 4](#)

[\[Notes en ligne\]](#)

Pour installer PHP sur des serveurs NT avec IIS 4, suivez les insctructions suivantes :

- Dans l'Internet Service Manager (MMC), sélectionnez le site web, ou le dossier racine.
- Ouvrez la feuille de propriétés du dossier (avec un clic droit dessus, puis en sélectionnant properties), puis cliquez dans l'onglet Home Directory, Virtual Directory, ou Directory.
- Cliquez dans le bouton Configuration, puis cliquez dans l'onglet App Mappings.
- Cliquez dans la bouton Add, et dans la boîte Executable, tapez: `c:\path-to-php-dir\php.exe %s %s`. Vous DEVEZ ajouter les `%s %s` à la fin, car sinon, PHP ne fonctionnera pas sans.
- Dans la boîte Extension, tapez le nom de l'extension de fichier que vous voulez associer à PHP. (Vous devez répéter les étapes 5 et 6 pour toutes les extensions que vous voulez associer à PHP. (.php et .phtml sont les plus répandus).
- Sélectionnez la sécurité appropriée (grâce à l'Internet Service Manager), et si votre serveur NT utilise NTFS, ajoutez les droits d'exécutions pour I_USR_ au dossier qui contient php.exe.

[4.4.4 Windows 9x/NT et Apache 1.3.x](#)

[\[Notes en ligne\]](#)

Vous devez éditer srm.conf ou httpd.conf pour configurer Apache, afin qu'il fonctionne avec PHP CGI. Bien qu'il puisse y avoir quelques variations de configurations de PHP sous Apache, elle est suffisamment simple pour être faite par un novice. Reportez vous à la doc Apache pour plus de détails.

- `ScriptAlias /php/ "c:/path-to-php-dir/"`
- `AddType application/x-httpd-php .php`
- `AddType application/x-httpd-php .phtml`
- `Action application/x-httpd-php "/php/php.exe"`

Pour utiliser la fonction de colorisation de la syntaxe, créez simplement un script PHP, et ajoutez le code suivant : `<?php show_source ("original_php_script.php"); ?>`. Substituez `original_php_script.php` avec le nom du fichier dont vous voulez voir le source. (c'est le seul moyen de faire cela). *Note:* Sous Win-Apache tous les back-slash (\) dans un chemin de fichier (tel que "c:\directory\file.ext"), doivent être convertis en slash (/).

[4.4.5 Omni HTTPd 2.0b1 pour Windows](#)

[\[Notes en ligne\]](#)

La méthode la plus simple pour configurer le serveur est :

- Step 1: Installez Omni server
- Step 2: Faites un clic-droit sur l'icone bleur d'OmniHTTPd, sur le bureau, et selectionnez Properties
- Step 3: Cliquez sur Web Server Global Settings
- Step 4: Dans l'onglet 'External', entrez: `virtual = .php | actual = c:\path-to-php-dir\php.exe`
- Step 5: Dans l'onglet Mime, entrez: `virtual = wwwserver/stdcgi | actual = .php`
- Step 6: Cliquez sur OK

Répez les étapes 2 à 6 pour chaque extension que vous voulez associer à PHP.

4.4.6 Modules PHP

[\[Notes en ligne\]](#)

php_calendar.dll	Fonctions de conversions calendaires
php_crypt.dll	Fonctions de cryptage
php_dbase.dll	Fonctions DBase
php_dbm.dll	Librairie d'émulation GDBM via Berkely DB2
php_filepro.dll	Lecture des bases filepro
php_gd.dll	Bibliothèque GD (pour les manipulations d'images)
php_hyperwave.dll	Fonctions HyperWave
php_imap4r2.dll	Fonctions IMAP 4
php_ldap.dll	Fonctions LDAP
php_mysql1.dll	Fonctions mSQL 1
php_mysql2.dll	Fonctions mSQL 2
php_mssql.dll	Fonctions MSSQL (requiert MSSQL DB-Libraries)
php3_mysql.dll (compilé dans PHP 4)	Fonctions MySQL
php_nsmail.dll	Fonctions Netscape mail
php_oci73.dll	Fonctions Oracle
php_snmp.dll	Fonctions SNMP get and walk (NT uniquement!)
php_zlib.dll	Fonctions ZLib

4.4.7 Installshield

[\[Notes en ligne\]](#)

L'installateur PHP pour Windows PHP, disponibles depuis les pages de téléchargement à <http://www.php.net/> installe la version CGI de PHP et, pour IIS, PWS, et Xitami, configure le serveur web en même temps.

Installez votre serveur HTTP sur votre système, puis assurez vous qu'il fonctionne.

Lancez l'installateur (.exe), et suivez les instructions fournies par le wizard. Deux types d'installations sont supportés : standard, qui effectue une configuration standard, et avancé, qui demande la configuration au fur et à mesure.

Le wizard d'installation rassemble suffisamment d'informations pour configurer le fichier `php.ini` et configurer le serveur web pour qu'il utilise PHP. Pour IIS et PWS sous NT Workstation, il affiche une liste de tous les noeuds du serveur, avec leur configuration. Vous pouvez alors choisir quels noeuds bénéficieront de la configuration PHP.

Une fois l'installation complète, l'installateur indiquera qu'il faut redémarrer votre système, et le fera pour voir. Ou bien, vous pourrez immédiatement utiliser PHP.

4.5 Problèmes?

[\[Notes en ligne\]](#)

4.5.1 Lisez la FAQ

[\[Notes en ligne\]](#)

Certains problèmes sont récurrents : Les plus commun sont listés dans la FAQ PHP, disponible à <http://www.php.net/FAQ.php>

4.5.2 Rapport de bug

[\[Notes en ligne\]](#)

Si vous pensez avoir trouvé un bug dans PHP, n'oubliez pas de le signaler. L'équipe de développement PHP ne le connaît peut être pas, et sans le signaler, vos chances de voir le bug corrigés sont nulles. Vous pouvez rapporter des bugs grâce au système de suivi, accessible à <http://www.php.net/bugs.php>.

4.5.3 Autres problèmes

[\[Notes en ligne\]](#)

Si vous êtes complètement bloqués, quelqu'un sur la liste de diffusion PHP pourra probablement vous aider. Essayez de consulter les archives, au cas où quelqu'un aurait déjà rencontré votre problème. Les archives sont toujours accessibles à : <http://www.php.net/>. Pour souscrire à la liste de diffusion, envoyez un mail vide à L'adresse de la mailing liste : php-general@lists.php.net.

Si vous voulez obtenir de l'aide sur la liste de diffusion PHP, essayez d'être concis et clair, et pensez à donner tous les détails sur votre environnement (OS, version de PHP, serveur web, CGI ou module, safe_mode...), et n'hésitez pas à envoyer suffisamment de code pour que nous puissions reproduire l'erreur.

5 Introduction

[\[Notes en ligne\]](#)

5.1 Qu'est ce que PHP?

[\[Notes en ligne\]](#)

PHP (officiellement "PHP: Hypertext Preprocessor") est un langage de script HTML, qui fonctionne coté serveur.

Réponse simple et claire, mais qu'est ce que cela veut dire? Un exemple :

Exemple d'introduction

```
<html>
  <head>
    <title>Example</title>
  </head>
  <body>
    <?php
echo "Hi, I'm a PHP script!";
    ?>
  </body>
</html>
```

Il est à noter la différence avec les autres scripts CGI écrit dans d'autres langages tels que le Perl ou le C : Au lieu d'écrire un programme avec de nombreuses lignes de commandes afin d'afficher une page HTML, vous écrivez une page HTML avec du code inclus à l'intérieur afin de réaliser une action précise (dans ce cas là, afficher du texte). Le code PHP est inclus entre [9.1.1 Le passage du HTML au PHP](#) qui permettent au navigateur de passer en "mode PHP".

Ce qui distingue le PHP des langages de script comme le Javascript est que le code est exécuté sur le serveur. Si vous avez un script similaire sur votre serveur, le client ne reçoit que le résultat du script, sans aucun moyen d'avoir accès au code qui a produit ce résultat. Vous pouvez configurer votre serveur web afin qu'il analyse tous vos fichiers HTML comme des fichiers PHP. Ainsi, il n'y a aucun moyen de distinguer les pages qui sont produites dynamiquement des pages statiques.

5.2 What can PHP do?

[\[Notes en ligne\]](#)

Le langage PHP possède les même fonctionnalités que les autres langages permettant d'écrire des scripts CGI, comme collecter des données, générer dynamiquement des pages web ou bien envoyer et recevoir des cookies.

La plus grande qualité et le plus important avantage du langage PHP est le support d'un grand nombre de bases de données. Réaliser une page web dynamique interagant une base de données est extrêmement simple. Les bases de données suivantes sont supportées par PHP:

- Adabas D

- dBase
- Empress
- FilePro (lecture seule)
- IBM DB2
- Informix
- Ingres
- InterBase
- FrontBase
- mSQL
- Direct MS–SQL
- MySQL
- ODBC
- Oracle (OCI7 et OCI8)
- PostgreSQL
- Solid
- Sybase
- Velocis
- Unix dbm

Le langage PHP inclut le support des services utilisant les protocoles tels que IMAP, SNMP, NNTP, POP3 ou encore HTTP. Vous pouvez également ouvrir des connexions et interagir en utilisant d'autres protocoles.

5.3 La genèse du PHP

[\[Notes en ligne\]](#)

Le langage PHP a été conçu durant l'automne 1994 par Rasmus Lerdorf. Les premières versions (qui restèrent privées) étaient utilisées afin de savoir qui venait consulter son CV en ligne. La première version publique fut disponible au début de l'année 1995. Elle fut connue sous le nom de "Personal Sommaire Page Tools". Elle était composée d'un analyseur extrêmement simple qui ne reconnaissait que quelques macros spéciales et d'un petit nombre d'utilitaires couramment utilisés dans les pages web. Un guestbook, un compteur, etc...

L'analyseur fut réécrit durant l'été 1995 et fut appelé PHP/FI Version 2. FI étaient les initiales d'un autre package que Rasmus avait écrit qui interprétait les formulaires HTML. C'est alors qu'il combina le "Personal Sommaire Page tools" avec le "Form Interpreter" et il y ajouta le support de mSQL: c'est comme cela que naquit PHP/FI. PHP/FI grandit de manière spectaculaire et de nombreuses personnes commencèrent à contribuer à son amélioration.

Il est relativement peu aisé de donner des statistiques, mais on estime que PHP/FI est utilisé sur 15 000 sites web dans le monde entier, fin 1996. Ce chiffre atteint 50 000 durant l'été 1997. L'été 1997 voit aussi un profond changement dans le développement du PHP: d'un projet personnel (celui de Rasmus),* on passe alors à un projet d'équipe. L'analyseur parseur fut de nouveau réécrit par Zeev Suraski et Andi Gutmans et ce nouvel analyseur forma la base de la version 3 du PHP. Une grande partie du code de PHP/FI fut complètement réécrit alors que l'autre partie fut portée pour donner le PHP Version 3.

Aujourd'hui (été 1999) PHP/FI ou PHP3 sont distribués avec de nombreux produits commerciaux comme "C2's StrongHold web server" et "RedHat Linux" et il est admis (d'après les chiffres de [NetCraft](#), et leurs [statistiques Netcraft Web Server Survey](#)) que le PHP est utilisé sur 150 000 sites web dans le monde entier. Pour comparaison, ce chiffre est supérieur au nombre de serveur tournant sous Netscape Enterprise server. Enfin, à l'heure où ce document est rédigé, la nouvelle génération du PHP est en cours de création. Elle utilisera les qualités de [Zend](#) pour améliorer les performances et améliorera le support des serveurs web autre que Apache.

6 Sécurité

[\[Notes en ligne\]](#)

PHP est un langage puissant et l'interpréteur, qu'il soit inclus dans le serveur web ou bien compilé en version CGI, est capable d'accéder aux fichiers, d'exécuter des commandes et d'ouvrir des connexions réseaux. Toutes ces propriétés rendent fragilise la sécurité d'un serveur web. Le langage PHP a été pensé afin d'être un langage beaucoup plus sécurisé pour écrire des *CGI* que le Perl ou le langage C. De plus une sélection rigoureuse des options de compilation et d'exécution vous permettront d'obtenir un équilibre parfait entre liberté et sécurité.

Etant donné qu'il y a de nombreux moyens d'utiliser le langage PHP, il y a de nombreuses directives de configuration afin d'en contrôler le comportement. Un grand nombre d'options permettent d'utiliser le PHP dans de nombreuses situations, mais cela signifie aussi qu'il y a certaines combinaisons d'options de compilation et d'exécution qui fragilise la sécurité du serveur. Ce chapitre explique comme les différentes options de configurations peuvent être combinées, tout en conservant une sécurité maximum.

6.1 CGI binary

[\[Notes en ligne\]](#)

6.1.1 Faiblesses connues

[\[Notes en ligne\]](#)

Utiliser le PHP comme un *CGI* exécutable vient la plupart du temps du fait que l'on ne veut pas l'utiliser comme un module du serveur web, (comme Apache), ou bien que l'on souhaite l'utiliser en combinaison d'un *CGI* complémentaire, afin de créer un environnement de script sécurisé (en utilisant des techniques de chroot ou setuid). Une telle décision signifie habituellement que vous installez votre exécutable dans le répertoire `cgi-bin` de votre serveur web. [CERT CA-96.11](#) recommande effectivement de placer l'interpréteur à l'intérieur du répertoire `cgi-bin`. Même si le binaire PHP peut être utilisé comme interpréteur indépendant, PHP a été pensé afin de rendre impossible les attaques que ce type d'installation induit.

- Accès au système de fichier: ``http://ma.machine/cgi-bin/php?/etc/passwd'`
Lorsque la requête est passée dans une url, après le point d'interrogation (?), elle est envoyée à l'interpréteur comme une ligne de commande par l'interface CGI. Habituellement, l'interpréteur ouvre le fichier spécifié et l'exécute.
Lorsqu'il est invoqué comme exécutable CGI, le PHP refuse d'interpréter les arguments de la ligne de commande.
- Accès d'un document web sur le serveur :
``http://my.host/cgi-bin/php/secret/doc.html'`
Le "path information" dans l'url, situé juste après le nom du binaire PHP, ``/secret/doc.html'` est utilisé par convention pour spécifier le nom du fichier qui doit être ouvert et interprété par le programme *CGI*. Habituellement, des directives de configuration du serveur web (pour le serveur Apache: Action) sont utilisées pour rediriger les requêtes pour obtenir un document ``http://my.host/secret/script.php3'` par l'interpréteur PHP. Dans une telle configuration, le serveur web vérifie d'abord si il a accès au répertoire ``/secret'`, et après cette vérification redirige la requête vers ``http://my.host/cgi-bin/php/secret/script.php3'`. Malheureusement, si la requête est faite directement sous cette forme, aucune vérification d'accès n'est faite par le serveur web pour le fichier ``/secret/script.php3'`, mais uniquement pour le fichier ``/cgi-bin/php'`. De cette manière, n'importe quel utilisateur qui peut accéder au fichier

`/cgi-bin/php' peut aussi accéder au document protégés sur le serveur web. Avec le PHP, l'option de compilation `@xref{enable-force-cgi-redirect}` et les options d'exécution [7.1.1.6 ini.doc-root](#) et [7.1.1.27 ini.user-dir](#) peuvent être utilisées pour prévenir ce genre d'attaques, si des restrictions d'accès sont appliquées sur les documents du serveur. Voir ci-dessous pour des explications plus complètes sur les différentes combinaisons.

6.1.2 Cas 1: Tous les fichiers sont publics

[\[Notes en ligne\]](#)

Si votre serveur n'a aucun document dont l'accès est restreint par un mot de passe ou un système de vérification de l'adresse IP, vous n'avez aucun besoin de ce type de configuration. Si votre serveur web ne permet pas les redirections, ou si votre serveur web n'a aucun besoin de communiquer avec le binaire PHP de manière sécurisée, vous pouvez utiliser l'option de compilation `@xref{enable-force-cgi-redirect}`. Vous devez quand même vérifier qu'aucun script ne fait appel au PHP, de manière directe, ``http://my.host/cgi-bin/php/dir/script.php3'` ou bien de manière indirecte, par redirection, ``http://my.host/dir/script.php3'`. Les redirections peuvent être configurées dans les fichiers de configuration d'Apache en utilisant les directives "AddHandler" et "Action" (voir ci-dessous).

6.1.3 Cas 2: Utilisation de la directive de compilation --enable-force-cgi-redirect

[\[Notes en ligne\]](#)

Cette option de compilation prévient quiconque d'appeler directement un script avec l'url ``http://my.host/cgi-bin/php/secretdir/script.php3'`. Dans ce cas là, PHP parsera le fichier uniquement si il y a eu redirection. Habituellement, le serveur web Apache réalise une redirection gr'ce aux directives suivantes :

```
Action php3-script /cgi-bin/php
AddHandler php3-script .php3
```

Cette option a uniquement été testée avec Apache et compte sur Apache pour affecter la variable d'environnement non-standart *REDIRECT_STATUS* pour les requêtes redirigées. Dans le cas où votre serveur web ne supporte pas le renseignement du PHP, pour savoir si la requête a été redirigée ou non, vous ne pouvez pas utiliser cette option de compilation. Vous devez alors utiliser une des autres manières pour utiliser la version binaire CGI du PHP, comme exposé ci-dessous.

6.1.4 Cas 3: Utilisation du "doc_root" ou du "user_dir"

[\[Notes en ligne\]](#)

Ajouter un contenu interactif dans votre serveur web, comme des scripts ou des exécutables, est souvent considéré comme une pratique non-sécurisée. Si, par erreur, le script n'est pas exécuté mais affiché comme une page HTML classique, il peut en résulter un vol de propriété intellectuelle ou des problèmes de sécurité à propos des mots de passe notamment. Donc, la plupart des administrateurs préfèrent mettre en place un répertoire spécial pour les scripts qui est uniquement accessible par le biais du binaire CGI du PHP, et donc, tous les fichiers de ce répertoire seront interprétés et non affichés tels quel.

Aussi, si vous ne pouvez pas utiliser la méthode présentée ci-dessus, il est nécessaire de mettre en place un répertoire "doc_root" différent de votre répertoire "document root" de votre serveur web.

Vous pouvez utiliser la directive [7.1.1.6 ini.doc-root](#) dans le [7.1 Le fichier de configuration](#), ou vous pouvez

affecter la variable d'environnement **PHP_DOCUMENT_ROOT**. Si cette variable d'environnement est affectée, le binaire **CGI** du PHP construira toujours le nom de fichier à ouvrir avec **doc_root** et le "path information" de la requête, et donc vous serez sûr qu'aucun script n'est exécuté en dehors du répertoire prédéfini. (à l'exception du répertoire désigné par la directive **user_dir** Voir ci-dessous).

Une autre option possible ici est la directive [7.1.1.27 ini.user-dir](#). Lorsque la directive n'est pas activée, seulement les fichiers contenues dans le répertoire **doc_root** peuvent être ouverts. Ouvrir un fichier possédant l'url `http://my.host/~user/doc.php3` ne correspond pas à l'ouverture d'un fichier sous le répertoire racine de l'utilisateur mais à l'ouverture du fichier `~user/doc.php3` sous le répertoire "doc_root" (oui, un répertoire comment par un tilde [~]).

Si la directive "user_dir" est activée à la valeur `public_php` par exemple, une requête du type `http://my.host/~user/doc.php3` ouvrira un fichier appelé `doc.php3` sous le répertoire appelé `public_php` sous le répertoire racine de l'utilisateur. Si le répertoire racine des utilisateurs est `/home/user`, le fichier exécuté sera `/home/user/public_php/doc.php3`.

user_dir et **doc_root** sont deux directives totalement indépendantes et donc vous pouvez contrôler l'accès au répertoire "document root" séparément des répertoires "user directory".

[6.1.5 Cas 4: L'exécutable PHP à l'extérieur de l'arborescence du serveur](#)

[\[Notes en ligne\]](#)

Une solution extrêmement sécurisée consiste à mettre l'exécutable PHP à l'extérieur de l'arborescence du serveur web. Dans le répertoire `/usr/local/bin`, par exemple. Le problème de cette méthode est que vous aurez à rajouter la ligne suivante :

```
#!/usr/local/bin/php
```

dans tous les fichiers contenant des tags PHP. Vous devrez aussi rendre le binaire PHP exécutable. Dans ce cas-là, traitez le fichier exactement comme si vous aviez un autre script écrit en Perl ou en sh ou en un autre langage de script qui utilise `#!` comme mécanisme pour lancer l'interpréteur lui-même. `itself`.

Pour que l'exécutable PHP prenne en compte les variables d'environnement **PATH_INFO** et **PATH_TRANSLATED** correctement avec cette configuration, vous devez utiliser l'option de compilation `@xref{enable-discard-path}`.

[6.2 Module Apache](#)

[\[Notes en ligne\]](#)

Lorsque le PHP est compilé en tant que module Apache, ce module hérite des permissions accordées à l'utilisateur faisant tourner Apache (par défaut, l'utilisateur "nobody").

7 Configuration

[\[Notes en ligne\]](#)

7.1 Le fichier de configuration

[\[Notes en ligne\]](#)

Le fichier de configuration (appelé `php3.ini` dans la version 3.0 du PHP, et simplement `php.ini` dans la version 4.0) est lu par le PHP au démarrage. Si vous avez compilé PHP en module, le fichier n'est lu qu'une seule fois, au lancement du démon HTTP. Pour la version *CGI* le fichier est lu à chaque invocation.

Lorsque vous utilisez le module Apache vous pouvez aussi changer les paramètres de configurations en utilisant les directives dans les fichiers de configuration d'Apache et dans les fichiers `.htaccess`. Dans la version 3.0, à chaque directive de configuration présente dans le fichier de configuration d'Apache correspond une directive de configuration dans le fichier `php3.ini` à l'exception des directives préfixées par "php3_".

Dans la version 4.0, il n'y a seulement que quelques directives dans le fichier de configuration d'Apache qui vous permettent de modifier la configuration de PHP.

`php_value` *name value*

- Cette directive affecte une valeur à la variable spécifiée.
`php_flag` *name on/off*
- Cette directive est utilisée pour activer ou désactiver une option.
`php_admin_value` *name value*
- Cette directive affecte une valeur à la variable spécifiée. La directive "Admin" ne peut être utilisée que dans le fichier de configuration d'Apache, et non dans un fichier `.htaccess`.
`php_admin_flag` *name on/off*
- Cette directive est utilisée pour activer ou désactiver l'option précédente.

Vous pouvez voir l'état de votre configuration en utilisant la fonction [phpinfo\(\)](#). Vous pouvez aussi accéder aux valeurs de votre configuration de manière individuelle en utilisant la fonction [get_cfg_var\(\)](#).

7.1.1 Directives de configuration générale

[\[Notes en ligne\]](#)

7.1.1.1 ini.asp-tags

[\[Notes en ligne\]](#)

`asp_tags` booléen

- Active l'utilisation des balises de type ASP `<% %>`, en plus des traditionnelles balises `<?php ?>`. Cela inclus l'utilisation du raccourcis `<%= $value %>`. Pour plus d'informations, reportez vous à [9.1.1 Le passage du HTML au PHP](#).

Note : Le support des balises ASP a été ajouté dans la version 3.0.4.

[7.1.1.2 ini.auto-append-file](#)

[\[Notes en ligne\]](#)

auto_append_file chaîne de caractères

- Spécifie le nom d'un fichier qui sera automatiquement ajouté après le fichier principal. Le fichier est inclus comme si il avait été appelé avec la fonction [include\(\)](#), donc [7.1.1.13 ini.include-path](#) est utilisé.

Le mot réservé NONE désactive l'auto-`appending`.

Note : Si le script s'arrête par la fonction [exit\(\)](#), *auto-append* ne fonctionnera pas.

[7.1.1.3 ini.auto-prepend-file](#)

[\[Notes en ligne\]](#)

auto_prepend_file chaîne de caractères

- Spécifie le nom d'un fichier qui sera automatiquement ajouté avant le fichier principal. Le fichier est inclus comme si il avait été appelé avec la fonction [include\(\)](#), donc [7.1.1.13 ini.include-path](#) est utilisé.

Le mot réservé NONE désactive l'auto-`appending`.

[7.1.1.4 ini.cgi-ext](#)

[\[Notes en ligne\]](#)

cgi_ext chaîne de caractères

- (NDT : aucune documentation n'est fournie).

[7.1.1.5 ini.display-errors](#)

[\[Notes en ligne\]](#)

display_errors booléen

- Cette directive détermine si les erreurs doivent être affichées à l'écran au format HTML ou non.

[7.1.1.6 ini.doc-root](#)

[\[Notes en ligne\]](#)

doc_root string

- Le dossier racine de PHP. Utilisée uniquement si elle est définie. Si PHP est configuré en [7.1.3.1 ini.safe-mode](#), aucun fichier en dehors de ce dossier ne sera accessible.

[7.1.1.7 ini.engine](#)

[\[Notes en ligne\]](#)

engine boolean

- Cette directive ne sert vraiment que si PHP est un module Apache. Elle sert aux sites qui veulent activer ou désactiver l'analyse des fichiers par PHP, dossier par dossier. En mettant `php3_engine off` au bon endroit, dans le fichier ``httpd.conf'`, PHP peut être activé ou désactivé.

[7.1.1.8 ini.error-log](#)

[\[Notes en ligne\]](#)

error_log string

- Nom du fichier où les erreurs seront enregistrées. Si la valeur spéciale syslog est utilisée, les erreurs sont envoyées au système standard d'historique. Sous UNIX, c'est syslog(3) et sous Windows NT c'est l'historique d'événements. L'historique système n'est pas supporté sous Windows 95.

[7.1.1.9 ini.error-reporting](#)

[\[Notes en ligne\]](#)

error_reporting integer

- Fixe le niveau d'erreur. Ce paramètre est un entier, représentant un champs de bits. Ajoutez les valeurs suivantes pour choisir le niveau que vous désirez :

valeur du bit	niveau choisi
1	erreurs normales
2	alertes normales
4	erreurs d'analyseur (parser errors)
8	alertes non critiques

Par défaut, la valeur est de 7 (erreurs normales, alertes normales et erreurs d'analyseur sont affichées).

[7.1.1.10 ini.open-basedir](#)

[\[Notes en ligne\]](#)

open_basedir string

- Limite l'espace où PHP peut ouvrir des fichiers.
Lorsqu'un script essaie d'ouvrir un fichier avec les fonctions fopen ou gzopen (par exemple), la localisation du fichier est vérifiée. Si ce fichier est hors du dossier cité dans cette directive, PHP refusera de l'ouvrir. Tous les liens symboliques sont résolus, et subissent aussi la restriction. La valeurs spéciale . indique que le dossier courant du script est utilisé comme open_basedir. Sous Windows, séparez les noms de dossiers par un point virgule (;). Sur les autres systèmes, séparez les noms de dossiers par des deux points (:). Lorsque PHP est un module Apache, la valeur de la directive open_basedir des dossiers parents sont automatiquement hérités par les fils.
Note : *Le support pour les dossiers multiples a été ajouté dans 3.0.7.*
La valeur par défaut est : libre accès à tous les fichiers.

[7.1.1.11 ini.gpc-order](#)

[\[Notes en ligne\]](#)

gpc_order chaîne de caractères

- Etablit l'ordre de préséance des méthodes GET/POST/COOKIE. Par défaut, cette directive est établie à "GPC". En affectant "GP" à cette directive, PHP ignorera les cookies, et écrasera toute méthode GET utilisée par une méthode POST avec des variables du même nom.

[7.1.1.12 ini.ignore-user-abort](#)

[\[Notes en ligne\]](#)

ignore_user_abort chaîne de caractères

- Désactivée par défaut. Si cette directive est activée, alors tous les scripts lancés iront jusqu'à leur terme, même si le client se déconnecte en plein milieu. Voir aussi la fonction [ignore_user_abort\(\)](#).

[7.1.1.13 ini.include-path](#)

[\[Notes en ligne\]](#)

include_path string

- Spécifie une liste de dossier où les fonctions [require\(\)](#), [include\(\)](#) et `fopen_with_path` (NDtraducteur : cette fonction semble avoir disparue). iront chercher les fichiers. Le format est le même que celui de la variable d'environnement *PATH* : une liste de dossiers, séparés par des deux points (:) sous UNIX, et des points virgules (;), sous Windows. *UNIX include_path*

```
include_path=.: /home/httpd/php-lib
```

Windows include_path

```
include_path=".;c:\www\phplib"
```

La valeur par défaut pour cette directive est ., c'est à dire le dossier courant.

[7.1.1.14 ini.isapi-ext](#)

[\[Notes en ligne\]](#)

isapi_ext string

- (Aucune documentation n'est fournie)

[7.1.1.15 ini.log-errors](#)

[\[Notes en ligne\]](#)

log_errors boolean

- Indique où les messages d'erreur générés doivent être écrits. Cette fonction est spécifique aux serveurs.

[7.1.1.16 ini.magic-quotes-gpc](#)

[\[Notes en ligne\]](#)

magic_quotes_gpc boolean

- Fixe le mode `magic_quotes` pour les opérations GPC (Get/Post/Cookie). Lorsque `magic_quotes` est activé, tous les caractères ' (guillemets simples), " (guillemets doubles), \ (backslash) et NUL sont échappés avec un backslash. Si `magic_quotes_sybase` fonctionne aussi, les guillemets simples seront

échappés avec un autre guillemet simple, plutôt qu'un backslash.

[7.1.1.17 ini.magic-quotes-runtime](#)

[\[Notes en ligne\]](#)

magic_quotes_runtime boolean

- Si *magic_quotes_runtime* est activé, toutes les fonctions qui retournent des données d'une source externe, y compris les bases de données et les fichiers texte, verront leur guillemets échappés avec un backslash. Si *magic_quotes_sybase* est aussi activé, les guillemets simples seront échappés avec un autre guillemet simple, plutôt qu'un backslash.

[7.1.1.18 ini.magic-quotes-sybase](#)

[\[Notes en ligne\]](#)

magic_quotes_sybase boolean

- Si *magic_quotes_sybase* est activé, les guillemets simples seront échappés avec un autre guillemet simple, plutôt qu'un backslash, si *magic_quotes_gpc* ou *magic_quotes_runtime* est activé.

[7.1.1.19 ini.max-execution-time](#)

[\[Notes en ligne\]](#)

max_execution_time integer

- Fixe le temps maximal d'exécution d'un script, en secondes. Cela permet d'éviter que des scripts en boucles infinies ne saturent le serveur.

[7.1.1.20 ini.memory-limit](#)

[\[Notes en ligne\]](#)

memory_limit entier

- Grâce à cette option, vous pouvez donner la quantité maximale de mémoire qu'un script peut allouer. Cela permet de réserver toute la mémoire d'un serveur à un seul script.

[7.1.1.21 ini.nsapi-ext](#)

[\[Notes en ligne\]](#)

nsapi_ext chaîne de caractères

- Aucune documentation n'est fournie.

[7.1.1.22 ini.short-open-tag](#)

[\[Notes en ligne\]](#)

short_open_tag booléen

- Active ou désactive l'utilisation des balises courtes, (<? ?>). Si vous voulez utiliser PHP et XML en même temps, vous devez désactiver cette option. Si cette option est désactivée, vous devez utiliser la forme longue des tags, (<?php ?>).

[7.1.1.23 ini.sql.safe-mode](#)

[\[Notes en ligne\]](#)

sql.safe_mode booléen

- Aucune documentation n'est fournie.

[7.1.1.24 ini.track-errors](#)

[\[Notes en ligne\]](#)

track_errors booléen

- Si cette option est activée, le dernier message d'erreur sera placé dans la variable globale `$php_errormsg`.

[7.1.1.25 ini.track-vars](#)

[\[Notes en ligne\]](#)

track_vars booléen

- Si cette option est activée, lors de l'appel des méthodes GET, POST et l'utilisation des cookies, les variables sont disponibles dans des tableaux associatifs globaux appelés respectivement `$HTTP_GET_VARS`, `$HTTP_POST_VARS` et `$HTTP_COOKIE_VARS`.

[7.1.1.26 ini.upload-tmp-dir](#)

[\[Notes en ligne\]](#)

upload_tmp_dir chaîne de caractères

- Indique le répertoire utilisé lors du chargement d'un fichier sur un serveur. Ce répertoire doit être accessible en lecture pour l'utilisateur qui lance le script PHP.

[7.1.1.27 ini.user-dir](#)

[\[Notes en ligne\]](#)

user_dir chaîne de caractères

- Répertoire où sont stockés les fichiers PHP dans le répertoire d'un utilisateur. Par exemple, `public_html`.

[7.1.1.28 ini.warn-plus-overloading](#)

[\[Notes en ligne\]](#)

warn_plus_overloading booléen

- Si cette option est activée, PHP émet un warning lorsque l'opérateur plus (+) est utilisé sur une chaîne de caractères. Cela permet de repérer plus facilement les scripts qui doivent être réécrits en utilisant l'opérateur de concaténation (.) plutôt que l'opérateur plus.

7.1.2 Configuration des directives concernant le mail

[\[Notes en ligne\]](#)

7.1.2.1 ini.smtp

[\[Notes en ligne\]](#)

SMTP chaîne de caractères

- Sous Windows, adresse IP ou nom que PHP doit utiliser pour envoyer du mail avec la fonction [mail\(\)](#).

7.1.2.2 ini.sendmail-from

[\[Notes en ligne\]](#)

sendmail_from chaîne de caractères

- Sous Windows, valeur du champs "From:" qui doit être utilisée lors de l'envoi de mail.

7.1.2.3 ini.sendmail-path

[\[Notes en ligne\]](#)

sendmail_path chaîne de caractères

- Localisation du programme de `sendmail`, habituellement ``/usr/sbin/sendmail'` ou ``/usr/lib/sendmail'`. `configure` essaye de repérer la présence de `sendmail` par lui même, et affecte ce résultat par défaut. En cas de problème de localisation, vous pouvez établir une nouvelle valeur par défaut ici.
Tout système n'utilisant pas `sendmail` doit établir cette directive à la valeur chemin du programme de substitution qui remplace le serveur de mail, si celui-ci existe, par exemple, [Qmail](#). Dans ce cas la, vous devez mettre: ``/var/qmail/bin/sendmail'`.

7.1.3 Directives de configuration du "Safe Mode"

[\[Notes en ligne\]](#)

7.1.3.1 ini.safe-mode

[\[Notes en ligne\]](#)

safe_mode booléen

- Cette directive active ou désactive l'option "safe mode". Lisez le chapitre [6 Sécurité](#) pour plus d'informations.

7.1.3.2 ini.safe-mode-exec-dir

[\[Notes en ligne\]](#)

safe_mode_exec_dir chaîne de caractères

- Si l'option "safe mode" est activée, [system\(\)](#) et les autres fonctions exécutant des programmes systèmes refusent de se lancer si ces programmes ne sont pas placés dans ce répertoire.

7.1.4 Directives de configuration de débbugage.

[\[Notes en ligne\]](#)

7.1.4.1 ini.debugger.host

[\[Notes en ligne\]](#)

debugger.host chaîne de caractères

- Adresse IP ou nom de l'hôte utilisé pour le déboggage.

7.1.4.2 ini.debugger.port

[\[Notes en ligne\]](#)

debugger.port chaîne de caractères

- Numéro du port utilisé pour le déboggage.

7.1.4.3 ini.debugger.enabled

[\[Notes en ligne\]](#)

debugger.enabled booléen

- Activation ou désactivation du debugger.

7.1.5 Directives de chargement des extensions

[\[Notes en ligne\]](#)

7.1.5.1 ini.enable-dl

[\[Notes en ligne\]](#)

enable_dl booléen

- Cette directive n'est réellement utile que dans le cas d'une compilation comme module Apache. Vous pouvez activer le chargement dynamique des extensions avec la fonction [dl\(\)](#), et cela de manière locale à chaque serveur virtuel ou à chaque répertoire.

La principale raison qui pousse à désactiver le chargement dynamique est un problème de sécurité. Lorsque le chargement dynamique est activé, il est possible d'ignorer les directives "safe_mode" ou "open_basedir".

Par défaut, il est possible d'utiliser le chargement dynamique, sauf lorsque la directive "safe_mode" est activée. En effet, il est alors impossible d'utiliser la fonction [dl\(\)](#).

7.1.5.2 ini.extension-dir

[\[Notes en ligne\]](#)

extension_dir chaîne de caractères

- Définit le répertoire dans lequel le PHP doit chercher les extensions lors du chargement dynamique.

[7.1.5.3 ini.extension](#)

[\[Notes en ligne\]](#)

extension chaîne de caractères

- Définit les extensions qui doivent être chargées lors du démarrage du PHP.

[7.1.6 MySQL Configuration Directives](#)

[\[Notes en ligne\]](#)

[7.1.6.1 ini.mysql.allow-persistent](#)

[\[Notes en ligne\]](#)

mysql.allow_persistent booléen

- Active ou désactive les connexions persistentes à la base de données MySQL.

[7.1.6.2 ini.mysql.default-host](#)

[\[Notes en ligne\]](#)

mysql.default_host chaîne de caractères

- Adresse par défaut du serveur, à utiliser lors de la connexion à un serveur MySQL, si aucun hôte n'est spécifié.

[7.1.6.3 ini.mysql.default-user](#)

[\[Notes en ligne\]](#)

mysql.default_user chaîne de caractères

- Utilisateur par défaut, à utiliser lors de la connexion à un serveur MySQL, si aucun utilisateur n'est spécifié.

[7.1.6.4 ini.mysql.default-password](#)

[\[Notes en ligne\]](#)

mysql.default_password chaîne de caractères

- Mot de passe par défaut, à utiliser lors de la connexion à un serveur MySQL, si aucun mot de passe n'est spécifié.

[7.1.6.5 ini.mysql.max-persistent](#)

[\[Notes en ligne\]](#)

mysql.max_persistent entier

- Nombre maximum de connexions persistantes à une base de donnée MySQL, par processus.

[7.1.6.6 ini.mysql.max-links](#)

[\[Notes en ligne\]](#)

mysql.max_links entier

- Nombre de connexion maximum à une base de donnée MySQL, par processus, incluant les connexions persistantes

[7.1.7 Directives de configuration mSQL](#)

[\[Notes en ligne\]](#)

[7.1.7.1 ini.msql.allow-persistent](#)

[\[Notes en ligne\]](#)

msql.allow_persistent booléen

- Active ou désactive les connexions persistentes à la base de données mSQL.

[7.1.7.2 ini.msql.max-persistent](#)

[\[Notes en ligne\]](#)

msql.max_persistent entier

- Nombre maximum de connexions persistantes à une base de donnée mSQL, par processus.

[7.1.7.3 ini.msql.max-links](#)

[\[Notes en ligne\]](#)

msql.max_links entier

- Nombre maximum de connexions à une base de donnée mSQL, par processus, incluant les connexions persistantes.

[7.1.8 Directives de configuration Postgres](#)

[\[Notes en ligne\]](#)

[7.1.8.1 ini.pgsql.allow-persistent](#)

[\[Notes en ligne\]](#)

pgsql.allow_persistent booléen

- Active ou désactive les connexions persistantes à la base de données Postgres.

[7.1.8.2 ini.pgsql.max-persistent](#)

[\[Notes en ligne\]](#)

pgsql.max_persistent entier

- Nombre maximum de connexions persistantes à une base de données Postgres, par processus.

[7.1.8.3 ini.pgsql.max-links](#)

[\[Notes en ligne\]](#)

pgsql.max_links entier

- Nombre maximum de connexions à une base de donnée Postgres, par processus, incluant les connexions persistantes.

[7.1.9 Directives de configuration Sybase](#)

[\[Notes en ligne\]](#)

[7.1.9.1 ini.sybase.allow-persistent](#)

[\[Notes en ligne\]](#)

sybase.allow_persistent booléen

- Active ou désactive les connexions persistentes à la base de données Sybase.

[7.1.9.2 ini.sybase.max-persistent](#)

[\[Notes en ligne\]](#)

sybase.max_persistent entier

- Nombre maximum de connexions persistantes à une base de données Sybase par processus.

[7.1.9.3 ini.sybase.max-links](#)

[\[Notes en ligne\]](#)

sybase.max_links entier

- Nombre maximum de connexions à une base de données Sybase, par processus, incluant les connexions persistantes.

[7.1.10 Sybase-CT Configuration Directives](#)

[\[Notes en ligne\]](#)

[7.1.10.1 ini.sybct.allow-persistent](#)

[\[Notes en ligne\]](#)

sybct.allow_persistent booléen

- Active ou désactive les connexions persistantes à la base de données Sybase-CT. Par défaut, cette option est activée.

[7.1.10.2 ini.sybct.max-persistent](#)

[\[Notes en ligne\]](#)

sybct.max_persistent entier

- Nombre maximum de connexions persistantes à une base de données Sybase-CT par processus. Par

défaut, cette option est à -1 , ce qui signifie que le nombre de connexion est illimité.

[7.1.10.3 ini.sybct.max-links](#)

[\[Notes en ligne\]](#)

sybct.max_links entier

- Nombre maximum de connexions à une base de données Sybase-CT, par processus, incluant les connexions persistantes. Par défaut, cette option est à -1 , ce qui signifie que le nombre de connexions est illimité.

[7.1.10.4 ini.sybct.min-server-severity](#)

[\[Notes en ligne\]](#)

sybct.min_server_severity entier

- Les messages en provenance du serveur d'un niveau d'erreur égal à *sybct.min_server_severity* seront considérés comme des alertes (warnings). Cette valeur peut être modifiée à l'intérieur du script en appelant la fonction *sybase_min_server_severity* (NDtraducteur : cette fonction semble ne pas exister). Par défaut, cette valeur vaut 10.

[7.1.10.5 ini.sybct.min-client-severity](#)

[\[Notes en ligne\]](#)

sybct.min_client_severity entier

- Les messages en provenance de la librairie client avec un niveau d'erreur égal ou supérieur à *sybct.min_client_severity* seront considérés comme des alertes. Cette valeur peut être modifiée à l'intérieur du script en appelant la fonction *sybase_min_client_severity* (NDtraducteur : cette fonction semble ne pas exister). Par défaut, cette valeur vaut 10, ce qui annule tout rapport d'erreur.

[7.1.10.6 ini.sybct.login-timeout](#)

[\[Notes en ligne\]](#)

sybct.login_timeout entier

- Délai de validité d'une tentative de connexion. Il est à noter que si *max_execution_time* est dépassé avant que la connexion n'expire, le script sera terminé avant le message d'erreur. Par défaut, cette valeur vaut 1 minute.

[7.1.10.7 ini.sybct.timeout](#)

[\[Notes en ligne\]](#)

sybct.timeout entier

- Temps maximum en secondes avant qu'une tentative de requête "select_db" ou "query" non aboutie renvoie une erreur. Il est à noter que si *max_execution_time* est dépassé avant que la requête n'expire, votre script sera terminé avant le message d'erreur. Par défaut, il n'y a pas de limite.

[7.1.10.8 ini.sybct.hostname](#)

[\[Notes en ligne\]](#)

sybct.hostname chaîne de caractères

- Nom de l'hôte à partir duquel vous vous connectez, afin d'être affiché par la fonction `sp_who` (NDtraducteur : cette fonction semble ne pas exister). Par défaut, cette valeur égale à 0.

[7.1.11 Directives de configuration Informix](#)

[\[Notes en ligne\]](#)

[7.1.11.1 ini.ifx.allow-persistent](#)

[\[Notes en ligne\]](#)

ifx.allow_persistent booléen

- Active les connexions persistantes à une base de données Informix.

[7.1.11.2 ini.ifx.max-persistent](#)

[\[Notes en ligne\]](#)

ifx.max_persistent entier

- Nombre maximum de connexions persistantes à une base de données Informix, par processus.

[7.1.11.3 ini.ifx.max-links](#)

[\[Notes en ligne\]](#)

ifx.max_links entier

- Nombre maximum de connexions à une base de données Informix par processus, en incluant les connexions persistantes.

[7.1.11.4 ini.ifx.default-host](#)

[\[Notes en ligne\]](#)

ifx.default_host chaîne de caractères

- Hôte par défaut où se connecter si aucun hôte n'est spécifié par les fonctions [ifx_connect\(\)](#) ou [ifx_pconnect\(\)](#).

[7.1.11.5 ini.ifx.default-user](#)

[\[Notes en ligne\]](#)

ifx.default_user chaîne de caractères

- Utilisateur par défaut si aucun utilisateur n'est spécifié par les fonctions [ifx_connect\(\)](#) ou [ifx_pconnect\(\)](#).

[7.1.11.6 ini.ifx.default-password](#)

[\[Notes en ligne\]](#)

ifx.default_password chaîne de caractères

- Mot de passe par défaut si aucun mot de passe n'est spécifié par les fonctions [ifx_connect\(\)](#) ou [ifx_pconnect\(\)](#).

[7.1.11.7 ini.ifx.blobinfile](#)

[\[Notes en ligne\]](#)

ifx.blobinfile booléen

- Lorsque cette option est activée, les colonnes de type "blob" seront retournées dans un fichier. Par défaut, elles seront retournées en mémoire. Il est possible de modifier dynamiquement cette valeur grâce à la fonction [ifx_blobinfile_mode\(\)](#).

[7.1.11.8 ini.ifx.textasvarchar](#)

[\[Notes en ligne\]](#)

ifx.textasvarchar booléen

- Lorsque cette option est activée, les colonnes de type "TEXT" seront retournées dans une chaîne de caractères. Par défaut, elles seront retournées en mémoire. Il est possible de modifier dynamiquement cette valeur grâce à la fonction [ifx_textasvarchar\(\)](#).

[7.1.11.9 ini.ifx.byteasvarchar](#)

[\[Notes en ligne\]](#)

ifx.byteasvarchar booléen

- Lorsque cette option est activée, les colonnes de type "BYTE" seront retournées dans une chaîne de caractères. Par défaut, elles seront retournées en mémoire. Il est possible de modifier dynamiquement cette valeur grâce à la fonction [ifx_textasvarchar\(\)](#).

[7.1.11.10 ini.ifx.charasvarchar](#)

[\[Notes en ligne\]](#)

ifx.charasvarchar booléen

- Lorsque cette option est activée, les espaces en fin de chaîne de caractères seront conservés lors d'une commande FETCH.

[7.1.11.11 ini.ifx.nullformat](#)

[\[Notes en ligne\]](#)

ifx.nullformat booléen

- Lorsque cette option est activée, les colonnes de valeur NULL seront retournées comme des chaînes de caractères vides. Il est possible de modifier dynamiquement cette valeur grâce à la fonction [ifx_nullformat\(\)](#).

[7.1.12 Directives de configuration pour les calculs mathématiques.](#)

[\[Notes en ligne\]](#)

[7.1.12.1 ini.bcmath.scale](#)

[\[Notes en ligne\]](#)

bcmath.scale entier

- Nombre de chiffres après la virgule pour toutes les fonctions de précision mathématique arbitraire.

[7.1.13 Directives de configuration du navigateur.](#)

[\[Notes en ligne\]](#)

[7.1.13.1 ini.browscap](#)

[\[Notes en ligne\]](#)

browscap chaîne de caractères

- Nom du fichier de descriptif des clients HTML. Voir aussi [get_browser\(\)](#).

[7.1.14 Directives de configuration du driver ODBC unifié](#)

[\[Notes en ligne\]](#)

[7.1.14.1 ini.uodbc.default-db](#)

[\[Notes en ligne\]](#)

uodbc.default_db chaîne de caractères

- Source de données ODBC à utiliser par défaut avec les fonctions [odbc_connect\(\)](#) ou [odbc_pconnect\(\)](#).

[7.1.14.2 ini.uodbc.default-user](#)

[\[Notes en ligne\]](#)

uodbc.default_user chaîne de caractères

- Nom d'utilisateur défaut avec les fonctions [odbc_connect\(\)](#) ou [odbc_pconnect\(\)](#).

[7.1.14.3 ini.uodbc.default-pw](#)

[\[Notes en ligne\]](#)

uodbc.default_pw chaîne de caractères

- Mot de passe par défaut dans les fonctions [odbc_connect\(\)](#) ou [odbc_pconnect\(\)](#).

[7.1.14.4 ini.uodbc.allow-persistent](#)

[\[Notes en ligne\]](#)

uodbc.allow_persistent booléen

- Cette option active ou désactive les connexions persistantes à la base de données, via le canal ODBC.

[7.1.14.5 ini.uodbc.max-persistent](#)

[\[Notes en ligne\]](#)

uodbc.max_persistent entier

- Nombre maximum de connexions persistantes autorisées à la base de données.

[7.1.14.6 ini.uodbc.max-links](#)

[\[Notes en ligne\]](#)

uodbc.max_links entier

- Nombre maximum de connexions (persistantes ou non), par processus, à la base de données.

8 Caractéristiques

[\[Notes en ligne\]](#)

8.1 Gestion des connexions

[\[Notes en ligne\]](#)

Note : *Les informations suivantes ne sont valables qu'à partir de la version 3.0.7.*

Le statut des connexions est conservé en interne par PHP. Il y a trois états possibles :

- 0 – NORMAL (normal)
- 1 – ABORTED (annulé)
- 2 – TIMEOUT (périmé)

Lorsqu'un script PHP est en cours d'exécution, son état est NORMAL. Si le client distant se déconnecte, le statut devient ABORTED. En général, une telle déconnexion provient d'un arrêt temporaire. Si la durée maximale d'exécution de PHP est dépassée, (voir [set_time_limit\(\)](#)), le script prend le statut TIMEOUT. Vous pouvez en outre, décider si vous voulez que la déconnexion d'un client provoque l'arrêt de votre script. Il est parfois pratique de terminer le script, même si le client n'est plus là pour recevoir les informations. Cependant, par défaut, le script sera interrompu, et terminé dès que le client se déconnecte. Ce comportement peut être modifié avec la directive `ignore_user_abort` dans le fichier ``php.ini`` ou bien avec la directive Apache **`ignore_user_abort`** du fichier Apache `.conf` ou avec la fonction [ignore_user_abort\(\)](#). Si vous ne demandez pas à PHP d'ignorer la déconnexion, et que l'utilisateur se déconnecte, le script sera terminé. La seule exception est si vous avez enregistré une fonction de fermeture, avec [register_shutdown_function\(\)](#). Avec une telle fonction, lorsque l'utilisateur interromp sa requête, à la prochaine exécution du script, PHP va s'apercevoir que le dernier script n'a pas été terminé, et il va déclencher la fonction de fermeture. Cette fonction sera aussi appelée à la fin du script, si celui-ci se termine normalement. Pour pouvoir avoir un comportement différent suivant l'état du script lors de sa finalisation, vous pouvez exécuter des commandes spécifiques à la déconnexion gr'ce à la commande [connection_aborted\(\)](#). Cette fonction retournera vrai si la connexion a été annulée.

Votre script peut aussi expirer après un laps de temps. Par défaut, le délai est de 30 secondes. Cette valeur peut être changée en utilisant la directive PHP **`max_execution_time`** dans le fichier ``php.ini`` ou avec la directive **`php3_max_execution_time`**, dans le fichier Apache `` .conf `` ou encore avec la fonction [set_time_limit\(\)](#). Lorsque le délai expire, le script est terminé, et comme pour la déconnexion du client, une fonction de finalisation sera appelée. Dans cette fonction, vous pouvez savoir si c'est le délai d'expiration qui a causé la fin du script, en appelant la fonction [connection_timeout\(\)](#). Cette fonction retournera vrai si le délai d'expiration a été dépassé.

Une chose à noter et que les deux cas ABORTED et TIMEOUT peuvent être appelés en même temps. Ceci est possible si vous demandez à PHP d'ignorer les déconnexions des utilisateurs. PHP va quand même noter le fait que l'utilisateur s'est déconnecté, mais le script va continuer. Puis, lorsqu'il atteint la limite de temps, le script va expirer. A ce moment là, les deux fonctions [connection_timeout\(\)](#) et [connection_aborted\(\)](#) vont retourner vrai. Vous pouvez aussi vérifier les deux états en un seul appel avec la fonction [connection_status\(\)](#). Cette fonction va retourner un champs de bits, avec les états. Si les deux états sont actifs, l'état retourné prendra la valeur 3.

8.2 Cookies

[\[Notes en ligne\]](#)

PHP supporte les cookies de manière transparente. Les cookies sont un mécanisme d'enregistrement d'informations sur le client, et de lecture de ces informations. Ce système permet d'authentifier et de suivre les visiteurs. Vous pouvez envoyer un cookie avec la commande [setcookie\(\)](#). Les cookies font partie de l'entête HTTP, ce qui impose que [setcookie\(\)](#) soit appelé avant tout affichage sur le client. Ce sont les mêmes limitations que pour [header\(\)](#).

Tous les cookies qui sont envoyés au client seront automatiquement retournés au script PHP, et transformés en variable, exactement comme pour GET et POST. Si vous souhaitez affecter plusieurs valeurs à un seul cookie, ajoutez[/] au nom du cookie. Pour plus de détails, reportez vous à la fonction [setcookie\(\)](#).

8.3 Gestion des erreurs

[\[Notes en ligne\]](#)

Il y a 4 types d'erreurs et d'alerte PHP :

- 1 – Erreur de fonction
- 2 – Alerte normale
- 4 – Erreur d'analyse
- 8 – Notes (alertes qui peuvent être ignorées mais qui peuvent avoir des conséquences sur le code)

Les 4 numéros attachés à un type d'erreur ont été ajoutés pour définir un niveau d'erreur. Par défaut, le niveau d'erreur accepté est de 7, c'est à dire 1 + 2 + 4, ou n'importe quoi, sauf les notes. Ce niveau peut être changé, dans le fichier php.ini avec la directive `error_reporting`. Il peut aussi être modifié dans le fichier de configuration Apache httpd.conf avec la directive `php3_error_reporting` ou bien encore, lors de l'exécution, en définissant la fonction [error_reporting\(\)](#).

Toutes les [9.4 Les expressions](#) peuvent être appelées avec le préfixe "@", ce qui permet d'ignorer le rapport d'erreur pour cette fonction particulière. Si une erreur survient dans une telle expression, et que l'option [7.1.1.24 ini.track-errors](#) est activée, vous pouvez retrouver le message d'erreur dans la variable globale `php_errormsg`.

8.4 Gestion des chargements de fichier

[\[Notes en ligne\]](#)

8.4.1 Chargements de fichiers par méthode POST

[\[Notes en ligne\]](#)

PHP est capable de recevoir des fichiers émis par un navigateur conforme à la norme RFC-1867 (c'est à dire Netscape Navigator 3 ou supérieur, Microsoft Internet Explorer 3 avec un patch de Microsoft, ou supérieur sans le patch). Cette fonctionnalité permet de charger des fichiers texte binaire. Avec l'authentification et les fonctions de manipulation des fichiers, vous avez un contrôle total sur le chargement et la gestion des fichiers chargés.

Notez bien que PHP supporte aussi le chargement par la méthode PUT comme dans le navigateur Netscape Composer et les clients Amaya du W3C. Reportez vous au chapitre sur le [8.4.4 Chargement par méthode PUT](#).

Un écran de chargement de fichiers peut être constitué en créant un formulaire de la manière suivante :
Formulaire de chargement de fichier

```
<FORM ENCTYPE="multipart/form-data" ACTION="_URL_" METHOD=POST>
<INPUT TYPE="hidden" name="MAX_FILE_SIZE" value="1000">
Send this file: <INPUT NAME="userfile" TYPE="file">
<INPUT TYPE="submit" VALUE="Send File">
</FORM>
```

Le paramètre `_URL_` doit pointer sur un fichier PHP. L'option `MAX_FILE_SIZE` cachée doit précéder le nom du fichier à charger, et représente la taille maximale du fichier à charger. La valeur est donnée en octets. Dans ce script, les valeurs suivantes doivent être définies pour assurer un chargement correct :

- `$userfile` – Le nom temporaire du fichier qui sera chargé sur la machine serveur.
- `$userfile_name` – Le nom du fichier original sur le système de l'expéditeur.
- `$userfile_size` – La taille envoyée en octets.
- `$userfile_type` – Le type mime du fichier, si le navigateur a fourni cette information. Par exemple, "image/gif".

Notez que `$userfile` prend la valeur qui est passée dans le champ `INPUT` de type `TYPE=file`. Dans l'exemple ci-dessus, nous avons choisi de l'appeler "userfile".

Les fichiers seront enregistrés par défaut dans le dossier temporaire du système. Ceci peut être changé en modifiant la variable d'environnement `TMPDIR` de l'environnement dans lequel PHP fonctionne. Vous pouvez modifier cette valeur depuis PHP même, avec la fonction [putenv\(\)](#).

Le script PHP qui reçoit le fichier chargé doit pouvoir gérer le fichier de manière appropriée. Vous pouvez utiliser la variable `$file_size` pour recalculer tous les fichiers qui sont trop gros ou trop petit. Vous pouvez utiliser la variable `$file_type` pour recalculer les fichiers qui n'ont pas le bon type. Quelque soit les actions, ce script doit pouvoir supprimer le fichier du dossier temporaire, ou le déplacer ailleurs.

Le fichier sera automatiquement effacé du dossier temporaire à la fin du script, si il n'a pas été déplacé ou renommé.

8.4.2 Erreurs classiques

[\[Notes en ligne\]](#)

La variable `MAX_FILE_SIZE` ne peut pas spécifier une taille de fichier plus grande que la taille qui a été fixée par `upload_max_filesize`, dans le fichier `PHP3.ini`, ou par `php3_upload_max_filesize` dans les directives Apache. La valeur par défaut est 2 Megaoctets.

Attention : il semble que CERN httpd supprime tout ce qui est après le premier caractère dans l'entête MIME. Tant que c'est le cas, CERN httpd ne pourra pas effectuer de chargement.

8.4.3 Chargement multiples de fichiers

[\[Notes en ligne\]](#)

Il est possible de charger plusieurs fichiers en même temps, et recevoir les informations adéquates organisées sous forme de tableau. Pour ce faire, il faut utiliser la même syntaxe d'envoi dans le code HTML que pour les sélections ou boîte à cocher multiples.

Note : *Le support du chargement multiple de fichier a été ajouté dans la version 3.0.10.*

Chargement multiple de fichier

```
<form action="file-upload.html" method="post" enctype="multipart/form-data">
Send these files:<br>
  <input name="userfile[]" type="file"><br>
  <input name="userfile[]" type="file"><br>
  <input type="submit" value="Send files">
</form>
```

Lorsque le formulaire ci dessus est envoyé, les tableaux `$userfile`, `$userfile_name`, et `$userfile_size` seront initialisés (ainsi que `$HTTP_POST_VARS`). Chaque tableau sera de type numérique, et contiendra les valeurs appropriées pour le chargement des fichiers.

Par exemple, si on veut envoyer les fichiers `~/home/test/review.html` et `~/home/test/xwp.out`, `$userfile_name[0]` contiendra `review.html`, et `$userfile_name[1]` contiendra `xwp.out`. De même, `$userfile_size[0]` contiendra la taille de `review.html`, `$userfile_size[1]` contiendra la taille de `xwp.out`, etc

8.4.4 Chargement par méthode PUT

[\[Notes en ligne\]](#)

PHP supporte la méthode HTTP PUT utilisée par les navigateurs tels que Netscape Composer et W3C Amaya. Les requêtes de type PUT sont beaucoup plus simples que les chargements de fichiers, et elles ressemblent à :

```
PUT /path/filename.html HTTP/1.1
```

Normalement, cela signifie que le client distant va sauver les données qui suivent dans le fichier `/path/filename.html` de votre disque. Ce n'est évidemment pas très sécurisé de laisser Apache ou PHP écraser n'importe quel fichier de l'arborescence. Pour éviter ceci, il faut d'abord dire au serveur que vous voulez qu'un script PHP donné gère la requête. Avec Apache, il y a une directive pour cela : *Script*. Elle peut être placée n'importe où dans le fichier de configuration d'Apache. En général, les webmasters la place dans le block `<Directory>`, ou peut être dans le bloc `<Virtualhost>`. La ligne suivante fera très bien l'affaire :

```
Script PUT /put.php3
```

Elle indique à Apache qu'il doit envoyer les requêtes de chargement par méthode PUT au script `put.php3`. Bien entendu, cela présuppose que vous avez activé PHP pour qu'il prenne en charge les fichiers de type `.php3`, et que PHP est actif.

Dans le fichier `put.php3` vous pouvez mettre ceci :

```
<? copy($PHP_UPLOADED_FILE_NAME,$DOCUMENT_ROOT.$REQUEST_URI); ?>
```

Ce script va copier le fichier chargé par le client distant à l'endroit désiré. Vous aurez probablement à effectuer quelques tests et des authentifications d'utilisateur, avant d'effectuer cette copie. Le seul piège est que lorsque PHP reçoit un chargement par méthode PUT, il va enregistrer le fichier dans le dossier temporaire, tout comme avec la [8.4.1 Chargements de fichiers par méthode POST](#). A la fin de la requête, le fichier sera effacé. Ce qui fait que ce script doit placer le fichier chargé quelque part. Le nom du fichier temporaire est placé dans la variable globale `$PHP_PUT_FILENAME`, et la destination prévue est placée

dans \$REQUEST_URI (ces noms peuvent changer d'une configuration d'Apache à l'autre). Cette destination est celle qui est demandée par le client, et vous n'avez pas à obéir aveuglément au client. Vous pourriez par exemple, déplacer le fichier dans un dossier de chargement.

8.5 Authentification HTTP avec PHP

[\[Notes en ligne\]](#)

Les fonctions d'authentification HTTP de PHP ne sont disponibles que si PHP est exécuté comme module Apache, et non pas sous la forme d'un CGI. Sous cette forme, il est possible d'utiliser la fonction [header\(\)](#) pour demander une authentification ("Authentication Required") au client, générant ainsi l'apparition d'une fenêtre de demande d'utilisateur et de mot de passe. Une fois que les champs ont été remplis, l'URL sera de nouveau appelée, avec les variables \$PHP_AUTH_USER, \$PHP_AUTH_PW et \$PHP_AUTH_TYPE contenant respectivement le nom d'utilisateur, le mot de passe et le type d'authentification. Actuellement, seule l'authentification simple ("Basic") est supportée. Reportez vous à la fonction [header\(\)](#) pour plus d'informations.

Voici un exemple de script qui force l'authentification du client pour accéder à une page : ***Exemple d'authentification HTTP***

```
<?php
if(!isset($PHP_AUTH_USER)) {
Header("WWW-Authenticate: Basic realm=\"My Realm\"");
Header("HTTP/1.0 401 Unauthorized");
echo "Texte à envoyer si le client appuie sur le bouton d'annulation\n";
exit;
} else {
echo "Bonjour $PHP_AUTH_USER.<P>";
echo "Vous avez entré le mot de passe $PHP_AUTH_PW.<P>";
}
?>
```

Au lieu d'afficher simplement les variables globales \$PHP_AUTH_USER et \$PHP_AUTH_PW, vous préférerez sûrement vérifier la validité du nom d'utilisateur et du mot de passe. Par exemple, en envoyant ces informations à une base de données, ou en recherchant dans un fichier dbm.

Méfiez vous des navigateurs buggés, tels que Internet Explorer. Ils semblent très suceptibles concernant l'ordre des entêtes. Envoyer l'entête d'authentification (*WWW-Authenticate*) avant le code de HTTP/1.0 401 semble lui convenir jusqu'à présent.

Pour éviter que quelqu'un écrive un script qui révèle les mots de passe d'une page, à la quelle on a accédé par une authentification traditionnelle, les variables globales PHP_AUTH ne seront pas assignées si l'authentification externe a été activée pour cette page. Dans ce cas, la variable \$REMOTE_USER peut être utilisée pour identifier l'utilisateur à l'extérieur.

Notez cependant que les manipulations ci-dessus n'empêchent pas quiconque qui possède une page non authentifiée de voler les mots de passes des pages protégées, sur le même serveur.

Netscape et Internet Explorer effaceront le cache d'authentification client si ils reçoivent une réponse 401. Cela permet de déconnecter un utilisateur, pour le forcer à ré-entrer son nom de compte et son mot de passe. Certains programmeurs l'utilisent pour donner un délai d'expiration, ou alors, fournissent un bouton de déconnexion.

Authentification HTTP avec nom d'utilisateur/mot de passe forcé

```

<?php
function authenticate() {
Header( "WWW-authenticate: basic realm='Test Authentication System'");
Header( "HTTP/1.0 401 Unauthorized");
echo "You must enter a valid login ID and password to access this resource\n";
exit;
}
if(!isset($PHP_AUTH_USER) || ($SeenBefore == 1 38; !strcmp($OldAuth, $PHP_AUTH_USER)) ) {
authenticate();
}
else {
echo "Welcome: $PHP_AUTH_USER<BR>";
echo "Old: $OldAuth";
echo "<FORM ACTION=\"\$PHP_SELF\" METHOD=POST>\n";
echo "<INPUT TYPE=HIDDEN NAME=\"SeenBefore\" VALUE=\"1\">\n";
echo "<INPUT TYPE=HIDDEN NAME=\"OldAuth\" VALUE=\"\$PHP_AUTH_USER\">\n";
echo "<INPUT TYPE=Submit VALUE=\"Re Authenticate\">\n";
echo "</FORM>\n";
}
?>

```

Ce comportement n'est pas nécessaire par le standard d'authentification HTTP Basic. Les tests avec Lynx ont montré qu'il n'affectait pas les informations de session lors de la réception d'un message de type 401, ce qui fait que passer ces informations entre le serveur et le client, et donnera l'accès à la ressource.

Notez aussi que tout ceci ne fonctionne pas sous Microsoft's IIS et que les limitations de PHP en version CGI sont dues aux limitations de IIS.

8.6 Création d'images

[\[Notes en ligne\]](#)

PHP n'est pas limité à la création de fichier HTML. Il peut aussi servir à créer des images JPG/PNG à la volée, aussi bien pour les émettre que pour les sauver. Il faut alors compiler PHP avec la librairie GD.

GIF creation with PHP

```

<?php
Header("Content-type: image/gif");
$string=implode($argv, " ");
$im = imagecreatefromgif("images/button1.gif");
$orange = ImageColorAllocate($im, 220, 210, 60);
$px = (imagesx($im)-7.5*strlen($string))/2;
ImageString($im, 3, $px, 9, $string, $orange);
ImageGif($im);
ImageDestroy($im);
?>

```

Cet exemple sera appelé depuis une page HTML avec un tag tel que: ``. Le script ci-dessus récupère le texte de la chaîne \$string et l'ajoute sur l'image de fond "images/button1.gif". Le résultat est alors envoyé au client. C'est un moyen très pratique d'éviter d'avoir à redessiner des boutons à chaque fois que le texte du bouton change. Avec ce script, il est généré dynamiquement.

8.7 Connexions persistantes aux bases de données

[\[Notes en ligne\]](#)

Les connexions persistantes aux bases de données SQL sont des connexions qui ne se referment pas à la fin du script. Lorsqu'une connexion persistante est demandée, PHP s'assure qu'il n'y a pas une autre connexion identique (qui serait ouverte précédemment, avec le même nom d'hôte, d'utilisateur et de mot de passe), et si une telle connexion existe, elle est utilisée. Sinon, elle est créée. Une connexion identique est une connexion qui a ouvert le même hôte, avec le même nom et même mot de passe (si ils sont nécessaires).

Ceux qui ne sont pas habitués aux techniques des serveurs web et leur distribution de la charge de travail, se font parfois une fausse idée de ces connexions persistantes. En particulier, cela ne permet par l'ouverture de plusieurs sessions avec le même lien, cela ne permet pas la réalisation de transactions efficaces, et cela ne fait pas le café. En fait, pour être extrêmement clair sur le sujet, les connexions persistantes ne vous donnent aucune fonctionnalité de plus qu'avec les connexions non persistantes.

Alors pourquoi?

Cela s'explique par la manière avec laquelle les serveurs web fonctionnent. Il y a trois manières d'utiliser PHP pour générer des pages.

La première est d'utiliser PHP comme un CGI (Common Interface Gateway). Lorsque PHP fonctionne de cette manière, une instance de l'interpréteur PHP est créée puis détruit pour chaque page demandée. Etant donné qu'il est détruit après chaque requête, toutes les ressources acquises (comme une connexion à une base SQL), sont purement et simplement détruites.

La deuxième méthode, et de loin, la plus prisée, est d'exécuter PHP sous la forme d'un module sur un serveur multi-process, ce qui revient à dire : Apache. Un tel serveur a typiquement un processus parent qui coordonne un ensemble de processus fils, qui servent les fichiers. Lorsque les requêtes parviennent depuis un client, elles sont transmises à un fils disponible. Cela signifie que si un client fait une deuxième requête, il peut être servi par un processus client différent du premier. Les connexions persistantes vous permettent alors de ne vous connecter à une base SQL que la première fois. Lors des connexions ultérieures, les processus fils pourront réutiliser la connexion ouverte précédemment.

La dernière méthode est d'utiliser PHP sous la forme d'un module. Pour un serveur multi-thread, actuellement, c'est purement théorique, car PHP ne fonctionne par encore sous cette forme. Le développement est en cours pour supporter ISAPI, WSAPI, et NSAPI (sous Windows), qui permettront d'utiliser PHP comme un module pour des serveurs tels que Netscape FastTrack, Microsoft's Internet Information Server (IIS), et O'Reilly's WebSite Pro. Lorsque cela sera fait, le comportement sera le même que pour les serveur multi-process.

Si les connexions persistantes n'ont aucune fonctionnalité de plus, à quoi servent-elles?

La réponse est extrêmement simple : efficacité. Les connexions persistantes sont un bon moyen d'accélérer les accès à une base SQL si le traitement de connexion à la base est long. Ce temps dépend de nombreux facteurs : le type de base de données, cette base est-elle sur le même serveur ou pas, quelle est la charge du serveur de base de données, etc... Si le temps de connexion est long, les connexions persistantes seront bien utiles, car une fois ouverte par un processus fils, la connexion est réutilisable sans avoir à se reconnecter. Si vous avez 20 processus fils, il suffit d'avoir 20 connexions persistantes ouvertes, une par fils.

Résumons nous : les connexions persistantes ont été définies pour avoir les mêmes fonctionnalités que les connexions non persistantes. Les deux types de connexions sont parfaitement interchangeables, et n'affecteront pas le comportement de votre script : uniquement son efficacité.

8.8 Utilisation des fichiers à distance

[\[Notes en ligne\]](#)

Aussi longtemps que le support de la fonction d'ouverture générique de fichiers ("URL fopen wrapper") est actif lorsque vous configurez PHP (il est inutile de passer explicitement l'option

—`disable-url-fopen-wrapper` pour faire la configuration), vous pouvez utiliser des URLs (HTTP et FTP) avec la plupart des fonctions qui utilisent un nom de fichier comme paramètre, ceci incluant les expressions [require\(\)](#) et [include\(\)](#). Note : *Vous ne pouvez pas utiliser les fichiers distants dans les expressions [include\(\)](#) et [require\(\)](#) avec Windows.*

Par exemple, vous pouvez suivre l'exemple suivant pour ouvrir un fichier sur un serveur web distant, analyser les résultats pour extraire les informations dont vous avez besoin, et ensuite l'utiliser dans une requête de base de données, ou simplement éditer les informations dans le style de votre site.

Connaître le titre du page distante

```
<?php
    $file = fopen("http://www.php.net/", "r");
    if (!$file) {
        echo "<p>Impossible d'ouvrir le fichier distant.\n";
        exit;
    }
    while (!feof($file)) {
        $line = fgets($file, 1024);
        /* Cela ne fonctionne que si le titre est écrit sur une ligne.*/
        if (eregi("<title>(.*?)</title>", $line, $out)) {
            $title = $out[1];
            break;
        }
    }
    fclose($file);
?>
```

Vous pouvez aussi écrire des fichiers sur un serveur FTP aussi longtemps que vous êtes connecté avec un utilisateur ayant les bons droits d'accès, alors que le fichier n'existait pas encore. Pour vous connecter avec un utilisateur autre qu'anonyme, vous devez spécifier un nom d'utilisateur (et certainement le mot de passe) dans l'URL, comme par exemple 'ftp://user:password@ftp.example.com/path/to/file'. (Vous pouvez utiliser le même type de syntaxe pour accéder aux fichiers via HTTP lorsqu'ils nécessitent une authentification basique.)

Stocker des données sur un serveur distant

```
<?php
    $file = fopen("ftp://ftp.php.net/incoming/outputfile", "w");
    if (!$file) {
        echo "<p>Impossible d'ouvrir un fichier distant en écriture.\n";
        exit;
    }
    /* Ecriture des données. */
    fputs($file, "$HTTP_USER_AGENT\n");
    fclose($file);
?>
```

Note : *Remarque: Vous pouvez avoir l'idée, à partir de l'exemple ci-dessus, d'utiliser la même technique pour écrire sur un log distant, mais comme mentionné ci-dessus vous ne pouvez qu'écrire sur un nouveau fichier en utilisant les fonctions [fopen\(\)](#) avec une URL. Pour faire des log distribués, nous vous conseillons de regarder la partie [syslog\(\)](#).*

9 Langage

[\[Notes en ligne\]](#)

9.1 La syntaxe de base

[\[Notes en ligne\]](#)

9.1.1 Le passage du HTML au PHP

[\[Notes en ligne\]](#)

Il y a quatre moyens pour passer du mode HTML au mode PHP :

Le passage du HTML au PHP

1. `<? echo ("Ceci est un exemple d'affichage à l'écran en PHP.\n"); ?>`
2. `<?php echo("Si vous voulez afficher du texte, faites comme ceci.\n"); ?>`
3. `<script language="php">`
`echo ("Certain éditeur HTML n'accepte pas les délimiteurs ci-dessus.");`
`</script>`
4. `<% echo ("Vous pouvez aussi utiliser le style ASP comme délimiteur."); %>`
`<%= $variable; # ceci est un raccourci pour "<%%echo .." %>`

La première possibilité n'est valable que si vous l'avez activée. Soit en faisant appel à la fonction `short_tags()` (NdT : semble avoir disparu), soit en utilisant l'option d'exécution [7.1.1.22 ini.short-open-tag](#) dans le fichier de configuration, soit en utilisant l'option de compilation `--enable-short-tags`.

La quatrième possibilité est seulement disponible si vous l'avez activée en utilisant soit l'option d'exécution [7.1.1.1 ini.asp-tags](#), soit en utilisant l'option de compilation `--enable-asp-tags compile-time`. Note : *Le support de la quatrième possibilité, ASP-style, a été ajouté dans la version 3.0.4.*

La parenthèse fermante pour un bloc ajoutera automatiquement un retour à la ligne si il y en a un de présent.

9.1.2 Le séparateur d'instruction

[\[Notes en ligne\]](#)

Les instructions sont séparées comme en C ou en Perl par un point virgule à chaque fin d'instruction.

Le tag de fin (`?>`) implique la fin d'une instruction, et donc ajoute implicitement un point virgule. Les deux exemples suivants sont équivalents.

```
<?php
echo "Ceci est un test";
?>
<?php echo "Ceci est un test" ?>
```

9.1.3 Comments

[\[Notes en ligne\]](#)

Le PHP supporte les commentaires comme en C, C++ et Shell Unix. Par exemple:

```
<?php
echo "Ceci est un test"; // Ceci est un commentaire sur une ligne comme en C++
    /* Ceci est un commentaire sur plusieurs lignes,
comme en C et C++ */
echo "Ceci est encore un test";
echo "Enfin, le test final"; # Ceci est un commentaire comme en Shell Unix
?>
```

Le premier type de commentaire ne commente que jusqu'à la fin de la ligne ou bien jusqu'à la fin du bloc, cela dépend du premier rencontré.

```
<h1>Ceci est un <?# echo "simple";?> exemple.</h1>
<p>La ligne du dessus affichera 'Ceci est un exemple'.
```

Faites attention à ne pas emboîter les commentaires de type 'C', ce qui arrive de temps en temps lorsque vous voulez commenter une grande partie de code.

```
<?php
/*
echo "Ceci est un test"; /* Ce commentaire va poser un problème */
*/
?>
```

9.2 Les constantes

[\[Notes en ligne\]](#)

Le PHP définit un certain nombre de constantes et propose des mécanismes pour en définir d'autres durant l'exécution. Les constantes se composent des variables, à l'exception du fait que leur valeur est définie gr'ce à la fonction [define\(\)](#), et qu'elle ne peut pas être modifiée par la suite.

Les constantes prédéfinies (toujours disponibles) sont :

`__FILE__`

- Le nom du fichier qui est actuellement exécuté. Si cette constante est utilisée dans le cadre d'un fichier "inclus" (ou require), alors le nom du fichier inclus est renvoyé, et non le nom du fichier parent.

`__LINE__`

- Le numéro de la ligne qui est actuellement exécutée. Si cette constante est utilisée dans le cadre d'un fichier "inclus" (ou require), c'est la position dans le fichier inclus qui est renvoyé.

`PHP_VERSION`

- La chaîne de caractères de présentation de la version du PHP qui est actuellement utilisée. Par exemple '4.0.0'.

`PHP_OS`

- Nom du système d'exploitation qui est utilisé par la machine qui fait tourner le PHP. Par exemple, 'Linux'.

`TRUE`

- La valeur TRUE.
FALSE
- La valeur FALSE.
E_ERROR
- Dénote une erreur autre qu'une "parsing error" (erreur d'analyse) qu'il n'est pas possible de corriger.
E_WARNING
- Dénote un contexte dans lequel le PHP trouve que quelque chose ne va pas. Mais l'exécution se poursuit tout de même. Ces alertes-là peuvent être récupérées par le script lui-même. Un exemple serait une expression régulière (regex) invalide dans la fonction [ereg\(\)](#).
E_PARSE
- L'analyseur a rencontré une forme syntaxique invalide dans le script. Correction de l'erreur impossible.
E_NOTICE
- Quelque chose s'est produit, qui peut être ou non une erreur. L'exécution continue. Par exemple, le cas de guillemets doubles (") non refermés, ou bien la tentative d'accéder à une variable qui n'est pas encore affectée.

Les constantes E_* sont généralement utilisées avec la fonction [error_reporting\(\)](#).

Vous pouvez définir d'autres constantes en utilisant la fonction [define\(\)](#).

Il est à noter que ce sont des constantes, et non pas des macros comme en C. Seulement les données scalaires peuvent être représentées par des constantes. *Définition de constantes*

```
<?php
define("CONSTANT", "Bonjour le monde.");
echo CONSTANT; // affiche "Bonjourle monde."
?>
```

Utilisatin des constantes __FILE__ et __LINE__

```
<?php
function report_error($file, $line, $message) {
echo "Une erreur a eu lieu dans le fichier $file à la ligne $line: $message.";
}
report_error(__FILE__, __LINE__, "Y a un problème!");
?>
```

9.3 Les structures de contrôle

[\[Notes en ligne\]](#)

Tous les scripts PHP sont une suite d'instructions. Une instruction peut être une assignation, un appel de fonction, une instruction conditionnelle ou bien une instruction qui ne fait rien (une instruction vide). Une instruction se termine habituellement par un point virgule (;). De plus, plusieurs instructions peuvent être regroupées en bloc, délimité par des accolades ("{}"). Un bloc est considéré comme une instruction. Les différents types d'instruction sont décrits dans ce chapitre.

[9.3.1 if](#)

[\[Notes en ligne\]](#)

L'instruction `if` est une des plus importantes instructions de tous les langages, PHP inclus. Elle permet l'exécution conditionnelle d'une partie de code. Les fonctionnalités de l'instruction `if` sont les mêmes en PHP qu'en C :

```
if (expr)
statement
```

Comme nous l'avons vu dans le paragraphe consacré aux expressions, *expr* est évaluée à sa vraie valeur. Si l'expression *expr* est TRUE, PHP exécutera l'instruction et si elle est FALSE, l'instruction sera ignorée. L'exemple suivant affiche la phrase `a est plus grand que b` si *\$a* est plus grand que *\$b*:

```
if ($a > $b)
print "a est plus grand que b";
```

Souvent, vous voulez que plusieurs instructions soient exécutées après un branchement conditionnel. Bien évidemment, il n'est pas obligatoire de répéter l'instruction conditionnelle autant de fois que vous avez d'instructions à exécuter. A la place, vous pouvez rassembler toutes les instructions dans un bloc. L'exemple suivant affiche `a est plus grand que b`, et assigne la valeur de la variable *\$a* à la variable *\$b*:

```
if ($a > $b) {
print "a est plus grand que b";
    $b = $a;
}
```

Vous pouvez imbriquer indéfiniment des instructions `if` les unes dans les autres, ce qui permet une grande flexibilité dans l'exécution d'une partie de code suivant un grand nombre de conditions.

[9.3.2 else](#)

[\[Notes en ligne\]](#)

Souvent, vous voulez exécuter une instruction si une condition est remplie, et une autre instruction si cette condition n'est pas remplie. C'est à cela que sert `else`. `else` fonctionne avec après un `if` et exécute les instructions correspondantes au cas où l'expression du `if` est FALSE. Dans l'exemple suivant, ce bout de code affiche `a est plus grand que b` si la variable *\$a* est plus grande que la variable *\$a*, et `a est plus petit que b` sinon:

```
if ($a > $b) {
print "a est plus grand que b";
} else {
print "a est plus petit que b";
}
```

Les instructions après le `else` ne sont exécutées que si l'expression du `if` est FALSE, et si elle n'est pas suivi par l'expression `elseif`.

[9.3.3 elseif](#)

[\[Notes en ligne\]](#)

elseif, comme son nom l'indique, est une combinaison de if et else. Comme l'expression else, il permet d'exécuter une instruction après un if dans le cas où le "premier" if est évalué comme FALSE. Mais, à la différence de l'expression else, il n'exécutera l'instruction que si l'expression conditionnelle elseif est évaluée comme TRUE. L'exemple suivant affichera a est plus grand que b, a est égal à b ou a est plus petit que b:

```
if ($a > $b) {
print "a est plus grand que b";
} elseif ($a == $b) {
print "a est égal à b";
} else {
print "a est plus petit que b";
}
```

Vous pouvez avoir plusieurs elseif qui s'imbriquent les uns dans les autres, après un if initial. Le premier elseif qui sera évalué à TRUE sera exécuté. En PHP, vous pouvez aussi écrire "else if" en deux mots et son comportement sera identique à la version en un seul mot.

L'expression elseif est exécutée seulement si le if précédent et tout autre elseif précédent est évalués comme FALSE, et que votre elseif est évalué à TRUE.

[9.3.4 Syntaxe alternative](#)

[\[Notes en ligne\]](#)

Le PHP propose une autre manière de rassembler des instructions à l'intérieur d'un bloc, pour les fonctions de contrôle if, while, for, et switch. Dans chaque cas, le principe est de remplacer l'accolade d'ouverture par deux points (:) et l'accolade de fermeture par, respectivement, endif;, endwhile;, endfor;, ou endswitch;.

```
<?php if ($a == 5): ?>
A is equal to 5
<?php endif; ?>
```

Dans l'exemple ci-dessus, le block HTML "A = 5" est inclus à l'intérieur d'un if en utilisant cette nouvelle syntaxe. Ce code HTML ne sera affiché que si la variable \$a est égale à 5.

Cette autre syntaxe fonctionne aussi avec le else et elseif. L'exemple suivant montre une structure avec un if, un elsif et un else utilisant cette autre syntaxe:

```
if ($a == 5):
print "a equals 5";
print "...";
elseif ($a == 6):
print "a equals 6";
print "!!!";
else:
print "a is neither 5 nor 6";
endif;
```

Allez voir [9.3.5 while](#), [9.3.7 for](#), et [9.3.1 if](#) pour d'autres exemples.

9.3.5 while

[\[Notes en ligne\]](#)

La boucle while est le moyen le plus simple d'implémenter une boucle en PHP. Cette boucle se comporte de la même manière qu'en C. L'exemple le plus simple d'une boucle while est le suivant :

```
while (expr) statement
```

La signification d'une boucle while est très simple. Le PHP exécute l'instruction tant que l'expression de la boucle while est évaluée comme TRUE. La valeur de l'expression est vérifiée à chaque début de boucle, et, si la valeur change durant l'exécution de l'instruction, l'exécution ne s'arrêtera qu'à la fin de l'itération (chaque fois que le PHP exécute l'instruction, on appelle cela une itération). De temps en temps, si l'expression du while est FALSE avant la première itération, l'instruction ne sera jamais exécutée.

Comme avec le if, vous pouvez regrouper plusieurs instructions dans la même boucle while en les regroupant à l'intérieur de parenthèses ou en utilisant la syntaxe suivante:

```
while (expr): statement ... endwhile;
```

Les exemples suivants sont identiques, et affichent tous les nombres de 1 à 10:

```
/* exemple 1 */
$i = 1;
while ($i <= 10) {
print $i++; /* La valeur affiche est $i avant l'incréméntation
              (post-incréméntation) */
}
/* exemple 2 */
$i = 1;
while ($i <= 10):
print $i;
    $i++;
endwhile;
```

9.3.6 do..while

[\[Notes en ligne\]](#)

Les boucles do..while ressemblent beaucoup aux boucles while, mais l'expression est testée à la fin de chaque itération plutôt qu'au début. La principale différence par rapport à la boucle while est que la première itération de la boucle do..while est toujours exécutée (l'expression n'est testée qu'à la fin de l'itération), ce qui n'est pas le cas lorsque vous utilisez une boucle while (l'expression est vérifiée au début de chaque itération).

Il n'y a qu'un syntaxe possible pour les boucles do..while:

```
$i = 0;
do {
print $i;
} while ($i>0);
```

La boucle ci-dessus ne va être exécutée qu'une seule fois, car lorsque l'expression est évaluée, elle vaut

FALSE (car la variable `$i` n'est pas plus grande que 0) et l'exécution de la boucle s'arrête.

Les utilisateurs familiers du C sont habitués à une utilisation différente des boucles `do..while`, qui permet de stopper l'exécution de la boucle au milieu des instructions, en l'encapsulant dans un `do..while(0)` la fonction [9.3.9 break](#). Le code suivant montre une utilisation possible:

```
do {
  if ($i < 5) {
    print "i n'est pas suffisamment grand";
    break;
  }
  $i *= $factor;
  if ($i < $minimum_limit) {
    break;
  }
  print "i est bon";
  ...process i...
} while(0);
```

Ne vous inquiétez pas si vous ne comprenez pas tout correctement. Vous pouvez écrire des scripts très très puissants sans utiliser cette fonctionnalité.

[9.3.7 for](#)

[\[Notes en ligne\]](#)

Les boucles `for` sont les boucles les plus complexes en PHP. Elles fonctionnent comme les boucles `for` du langage C. La syntaxe des boucles `for` est la suivante:

```
for (expr1; expr2; expr3) statement
```

La première expression (*expr1*) est évaluée (exécutée) quoi qu'il arrive au début de la boucle.

Au début de chaque itération, l'expression *expr2* est évaluée. Si l'évaluation vaut TRUE, la boucle continue et l'instruction est exécutée. Si l'évaluation vaut FALSE, l'exécution de la boucle s'arrête.

A la fin de chaque itération, l'expression *expr3* est évaluée (exécutée).

Les expressions peuvent éventuellement être laissées vides. Si l'expression *expr2* est laissée vide, cela signifie que c'est une boucle infinie (PHP considère implicitement qu'elle vaut TRUE, comme en C). Cela n'est pas vraiment très utile, à moins que vous souhaitiez terminer votre boucle par l'instruction conditionnelle [9.3.9 break](#).

Considérons les exemples suivants. Tous affichent les chiffres de 1 à 10:

```
/* exemple 1 */
for ($i = 1; $i <= 10; $i++) {
  print $i;
}
/* exemple 2 */
for ($i = 1;;$i++) {
  if ($i > 10) {
    break;
  }
  print $i;
}
/* exemple 3 */
$i = 1;
for (;;) {
```

```

if ($i > 10) {
    break;
}
print $i;
    $i++;
}
/* exemple 4 */
for ($i = 1; $i <= 10; print $i, $i++) ;

```

Bien évidemment, le premier exemple est le plus simple de tous (ou peut être le quatrième), mais vous pouvez aussi penser qu'utiliser une expression vide dans une boucle for peut être utile parfois. PHP supporte aussi la syntaxe alternative suivante pour les boucles for :

```
for (expr1; expr2; expr3): statement; ...; endfor;
```

Les autres langages ont l'instruction foreach pour accéder aux éléments d'un tableau. PHP3 ne dispose pas d'une telle fonction; PHP4 en dispose (voir [9.3.8 foreach](#)). En PHP3, vous pouvez combiner [9.3.5 while](#) avec [list\(\)](#) et [each\(\)](#) pour obtenir le même résultat. Reportez vous aux exemples de la documentation.

[9.3.8 foreach](#)

[\[Notes en ligne\]](#)

PHP4 (mais pas PHP3) inclus une commande foreach, comme en Perl ou d'autres langages. C'est un moyen simple de passer en revue un tableau. Il y a deux syntaxes possibles : la seconde est une extension mineure mais pratique de la première:

```

foreach(array_expression as $value) statement
foreach(array_expression as $key => $value) statement

```

La première forme passe en revue le tableau `array_expression`. A chaque itération, la valeur de l'élément courant est assigné à `$value` et le pointeur interne de tableau est avancé d'un élément (ce qui fait qu'à la prochaine itération, on accédera à l'élément suivant).

La deuxième forme fait exactement la même chose, mais c'est la clé de l'élément courant qui est assigné à la variable `$key`.

Lorsque foreach démarre, le pointeur interne de fichier est automatiquement ramené au premier élément du tableau. Cela signifie que vous n'aurez pas à faire appel à [reset\(\)](#) avant foreach.

Vous pouvez remarquer que les exemples suivants fonctionnent de manière identique :

```

reset ($arr);
while (list(, $value) = each ($arr)) {
    echo "Valeur: $value<br>\n";
}
foreach ($arr as $value) {
    echo "Valeur: $value<br>\n";
}

```

Les exemples suivants sont aussi fonctionnellement identiques :

```

reset ($arr);
while (list($key, $value) = each ($arr)) {
    echo "Clé: $key; Valeur: $value<br>\n";
}

```

```

}
foreach ($arr as $key => $value) {
echo "Clé: $key; Valeur: $value<br>\n";
}

```

Voici quelques exemples de plus :

```

/* exemple 1: valeur seule */
$a = array (1, 2, 3, 17);
foreach ($a as $v) {
print "Valeur courante de \$a: $v.\n";
}
/* exemple 1: valeur (avec clé associée) */
$a = array (1, 2, 3, 17);
$i = 0; /* pour affichage seulement*/
foreach($a as $v) {
print "\$a[$i] => $k.\n";
}
/* exemple 1: valeur et clé */
$a = array (
    "one" => 1,
    "two" => 2,
    "three" => 3,
    "seventeen" => 17
);
foreach($a as $k => $v) {
print "\$a[$k] => $v.\n";
}

```

[9.3.9 break](#)

[\[Notes en ligne\]](#)

L'instruction `break` permet de sortir d'une structure `if`, `for`, `while`, ou `switch`.

`break` accepte un argument numérique optionnel qui vous indiquera combien de structures emboîtées ont été interrompues.

```

$i = 0;
while ($i < 10) {
if ($arr[$i] == "stop") {
break; /* Vous pouvez aussi écrire 'break 1;' ici. */
}
    $i++;
}
/* Utilisation de l'argument optionnel. */
$i = 0;
while ( ++$i ) {
switch ( $i ) {
case 5:
echo "à 5<br>\n";
break 1; /* Ne sort que du switch. */
case 10:
echo "à 10; quitting<br>\n";
break 2; /* Sort du switch et du while. */
default:
break;
}
}

```

```

    }
}

```

9.3.10 continue

[\[Notes en ligne\]](#)

L'instruction continue est utilisée dans une boucle afin d'éviter les instructions de l'itération courante afin de passer directement à l'itération suivante.

continue accepte un argument numérique optionnel qui vous indiquera combien de structures emboîtées ont été ignorées.

```

while (list ($cle, $valeur) = each ($arr)) {
if (!($cle % 2)) { // évite les membres impairs
continue;
}
fonction_quelconque($valeur);
}
$i = 0;
while ($i++ < 5) {
echo "Dehors<br>\n";
while (1) {
echo " Milieu<br>\n";
while (1) {
echo " Intérieur<br>\n";
continue 3;
}
}
echo "Ceci n'est jamais atteint.<br>\n";
}
echo "Ceci non plus.<br>\n";
}

```

9.3.11 switch

[\[Notes en ligne\]](#)

L'instruction switch équivaut à une série d'instructions if. En de nombreuses occasions, vous aurez besoin de comparer la même variable (ou expression) avec un grand nombre de valeurs différentes, et d'exécuter différentes parties de code suivant la valeur à laquelle elle est égale. C'est exactement à cela que sert l'instruction switch.

Les deux exemples suivants sont deux manières différentes d'écrire la même chose, l'une en utilisant une série de if, et l'autre en utilisant l'instruction switch:

```

if ($i == 0) {
print "i égale 0";
}
if ($i == 1) {
print "i égale 1";
}
if ($i == 2) {
print "i égale 2";
}
switch ($i) {

```

```

case 0:
print "i égale 0";
break;
case 1:
print "i égale 1";
break;
case 2:
print "i égale 2";
break;
}

```

Il est important de comprendre que l'instruction switch exécute chacune des clauses dans l'ordre. L'instruction switch est exécutée ligne par ligne. Au début, aucun code n'est exécuté. Seulement lorsqu'un case est vérifié, PHP exécute alors les instructions correspondantes. PHP continue d'exécuter les instructions jusqu'à la fin du bloc d'instructions du switch, ou bien dès qu'il trouve l'instruction break. Si vous ne pouvez pas utiliser l'instruction break à la fin de l'instruction case, PHP continuera à exécuter toutes les instructions qui suivent. Par exemple :

```

switch ($i) {
case 0:
print "i égale 0";
case 1:
print "i égale 1";
case 2:
print "i égale 2";
}

```

Dans cet exemple, si \$i est égal à 0, PHP va exécuter quand même toutes les instructions qui suivent. Si \$i est égal à 1, PHP exécutera les deux dernières instructions. Et seulement si \$i est égal à 2, vous obtiendrez le résultat escompté, c'est-à-dire, l'affiche de "i égal 2". Donc, l'important est de ne pas oublier l'instruction break (même si il est possible que vous l'omettiez dans certaines circonstances).

Dans une commande switch, une condition n'est évaluée qu'une fois, est le résultat est comparé à chaque case. Dans une structure elseif, les conditions sont évaluées à chaque comparaisons. Si votre condition est plus compliquée qu'une simple comparaison, ou bien fait partie d'une boucle, switch sera plus rapide.

La liste de commande d'un case peut être vide, auquel cas PHP utilisera la liste de commande du cas suivant.

```

switch ($i) {
case 0:
case 1:
case 2:
print "i est plus petit que 3 mais n'est pas négatif";
break;
case 3:
print "i égale 3";
}

```

Un case spécial est default. Ce cas est utilisé lorsque tous les case ont échoués. Par exemple :

```

switch ($i) {
case 0:
print "i égale 0";
break;
case 1:
print "i égale 1";
}

```

```
break;
case 2:
print "i égale 2";
break;
default:
print "i n'est ni égal à 2, ni à 1, ni à 0.";
}
```

Une autre chose à mentionner est que l'instruction `case` peut être une expression à de type scalaire, c'est-à-dire nombre entier, nombre à virgule flottante et chaîne de caractère. Les tableaux sont sans intérêt dans ce contexte-là.

La syntaxe alternative pour cette structure de contrôle est la suivante : (Pour plus d'informations, voir [9.3.4 Syntaxe alternative](#)).

```
switch ($i):
case 0:
print "i égale 0";
break;
case 1:
print "i égale 1";
break;
case 2:
print "i égale 2";
break;
default:
print "i n'est ni égal à 2, ni à 1, ni à 0";
endswitch;
```

9.3.12 [require\(\)](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

La commande [require\(\)](#) se remplace elle-même par le contenu du fichier spécifié, comme les préprocesseurs C le font avec la commande `#include`.

Il est important de noter que lorsqu'un fichier est [include\(\)](#) ou [require\(\)](#), les erreurs d'analyse apparaîtront en HTML tout au début du fichier, et l'analyse du fichier parent ne sera pas interrompue. Pour cette raison, le code qui est dans le fichier doit être placé entre [9.1.1 Le passage du HTML au PHP](#).

[require\(\)](#) n'est pas vraiment une fonction PHP : c'est plus une instruction du langage. Elle ne fonctionne pas comme les fonctions standards. Par exemple, [require\(\)](#) ne peut pas contenir d'autres structures de contrôle. De plus, il ne retourne aucune valeur. Lire une valeur retournée par un [require\(\)](#) retourne une erreur d'analyse. Contrairement à [include\(\)](#), [require\(\)](#) va *toujours* lire dans le fichier cible, même si la ligne n'est jamais exécutée. Si vous souhaitez une inclusion conditionnelle, utilisez [include\(\)](#). La condition ne va jamais affecter [require\(\)](#). Cependant, si la ligne de [require\(\)](#) n'est jamais exécutée, le code du fichier ne le sera jamais non plus.

Les boucles n'affectent pas le comportement de [require\(\)](#). Même si le code contenu dans le fichier source est appelé dans la boucle, [require\(\)](#) n'est exécuté qu'une fois.

Cela signifie qu'on ne peut pas mettre un [require\(\)](#) dans une boucle, et s'attendre à ce qu'il inclue du code à chaque itération. Pour cela, il faut utiliser [include\(\)](#).

```
require ('header.inc');
```

Attention : [include\(\)](#) et [require\(\)](#) ajoute le contenu du fichier cible dans le script lui-même. Elle n'utilise pas le protocole HTTP ou tout autre protocole. Toute variable qui est dans le champs du script sera accessible dans le fichier d'inclusion, et vice versa.

```
require ("file.inc?varone=1&vartwo=2"); /* Ne fonctionne pas. */
$varone = 1;
$vartwo = 2;
require ("file.inc"); /* $varone et $vartwo seront accessible à file.inc */
```

Ne vous laissez pas abuser par le fait que vous pouvez requérir ou inclure des fichiers via HTTP en utilisant la fonctionnalité de [8.8 Utilisation des fichiers à distance](#) ce qui est au dessus reste vrai.

En PHP3, il est possible d'exécuter une commande return depuis un fichier inclus, tant que cette commande intervient au niveau global du fichier inclus. Elle ne doit intervenir dans aucun bloc (entre accolade {}). En PHP4, cette possibilité a été supprimée. Si vous en avez besoin, utilisez plutôt [include\(\)](#).

9.3.13 [include\(\)](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

La fonction [include\(\)](#) inclus et évalue le fichier spécifié en argument.

Il est important de noter que lorsqu'un fichier est [include\(\)](#) ou [require\(\)](#), les erreurs d'analyse apparaîtront en HTML tout au début du fichier, et l'analyse du fichier parent ne sera pas interrompue. Pour cette raison, le code qui est dans le fichier doit être placé entre [9.1.1 Le passage du HTML au PHP](#).

Cela a lieu à chaque fois que la fonction [include\(\)](#) est rencontrée. Donc vous vous pouvez utiliser la fonction [include\(\)](#) dans une boucle pour inclure un nombre infini de fois un fichier, ou même des fichiers différents.

```
$files = array ('first.inc', 'second.inc', 'third.inc');
for ($i = 0; $i < count($files); $i++) {
include $files[$i];
}
```

[include\(\)](#) diffère de [require\(\)](#) car le fichier inclus est ré-évaluée à chaque fois que la commande est exécutée, tandis que [require\(\)](#) est remplacée par le fichier cible lors de la première exécution, que son contenu soit utilisé ou non. De plus, cela se fait même si il est placé dans une structure conditionnelle, comme dans un [9.3.1 if](#)).

Parce que la fonction [include\(\)](#) nécessite une construction particulière, vous devez l'inclure dans un bloc si elle est incluse dans une structure conditionnelle.

```
/* Ceci est faux, et ne fonctionnera pas ce qu'on attend. */
if ($condition)
include($file);
else
include($other);
/* Ceci est CORRECT. */
if ($condition) {
include($file);
} else {
include($other);
}
```

En PHP3, il est possible d'exécuter une commande return depuis un fichier inclus, tant que cette commande intervient au niveau global du fichier inclus. Elle ne doit intervenir dans aucun bloc (entre accolade {}). En

PHP4, cette possibilité a été supprimée. Cependant, PHP4 vous autorise à retourner des valeurs d'un fichier inclus. Vous pouvez traiter [include\(\)](#) comme une fonction normale, qui retourne une valeur. Mais cela génère une erreur d'analyse en PHP3.

[include\(\)](#) en php3 et php4

On suppose que le fichier `test.inc` existe, et est placé dans le même dossier que le fichier principal :

```
<?php
echo "Avant le retour<br>\n";
if (1) {
return 27;
}
echo "Après le retour <br>\n";
?>
```

On suppose que le fichier `main.html` contient ceci :

```
<?php
$retval = include ('test.inc');
echo "Fichier inclus: '$retval'<br>\n";
?>
```

Lorsque `main.html` est appelé en PHP3, il va générer une erreur d'analyse (parse error) à la ligne 2; vous ne pouvez pas vous attendre à un retour sur une fonction [include\(\)](#) en PHP3. En PHP4, cependant, le résultat sera :

```
Avant le retour
Fichier inclus : '27'
```

Supposons maintenant que `main.html` a été modifié et contient maintenant le code suivant :

```
<?php
include ('test.inc');
echo "Retour dans le main.html<br>\n";
?>
```

En PHP4, l'affichage sera :

```
Avant le retour
Retour dans le main.html
```

Au contraire, PHP3 affichera :

```
Avant le retour
27Retour dans le main.html
Parse error: parse error in /home/torben/public_html/phptest/main.html on line 5
```

L'erreur d'analyse ci-dessus est le résultat du fait que la commande return est dans un bloc qui n'est pas une

fonction, dans `test.inc`. Lors que le return est sorti du bloc, l'affichage devient :

```
Avant le retour
27Retour dans le main.html
```

Le '27' est du au fait que PHP3 ne supporte pas le return dans ces fichiers.

Il est important de noter que lorsqu'un fichier est [include\(\)](#) ou [require\(\)](#), les erreurs d'analyse apparaîtront en HTML tout au début du fichier, et l'analyse du fichier parent ne sera pas interrompue. Pour cette raison, le code qui est dans le fichier doit être placé entre [9.1.1 Le passage du HTML au PHP](#).

```
include ("file.inc?varone=1&vartwo=2"); /* ne fonctionne pas. */
$varone = 1;
$vartwo = 2;
include ("file.inc"); /* $varone et $vartwo sont accessibles dans file.inc */
```

Ne vous laissez pas abuser par le fait que vous pouvez requérir ou inclure des fichiers via HTTP en utilisant la fonctionnalité de [8.8 Utilisation des fichiers à distance](#) ce qui est au dessus reste vrai.

Voir aussi [readfile\(\)](#), [require\(\)](#) et [virtual\(\)](#).

9.4 Les expressions

[\[Notes en ligne\]](#)

Les expressions sont la partie la plus importante du PHP. En PHP, presque tout ce que vous écrivez est une expression. La manière la plus simple de définir une expression est : "tout ce qui a une valeur".

Les formes les plus simples d'expressions sont les constantes et les variables. Lorsque vous écrivez "\$a = 5", vous assignez la valeur '5' à la variable \$a. Bien évidemment, '5' vaut 5 ou, en d'autres termes, '5' est une expression avec pour valeur 5 (dans ce cas, '5' est un entier constant).

Après cette assignation, vous pouvez considérer que \$a a pour valeur 5 et donc, écrire \$b = \$a, revient à écrire \$b = 5. En d'autres termes, \$a est une expression avec de valeur 5. Si tout fonctionne correctement, c'est exactement ce qui arrive.

Un exemple plus complexe concerne les fonctions. Par exemple, considérons la fonction suivante :

```
function foo () {
return 5;
}
```

Considérant que vous êtes familier avec le concept de fonction, (si ce n'est pas le cas, jetez un oeil au chapitre concernant les fonctions), vous serez d'accord que \$c = foo() est équivalent à \$c = 5, et vous aurez tout à fait raison. Les fonctions sont des expressions qui ont la valeur de leur "valeur de retour". Si foo() renvoie 5, la valeur de l'expression 'foo()' est 5. Habituellement, les fonctions ne font pas que renvoyer une valeur constante mais réalisent des traitements.

Bien sur, les valeurs en PHP n'ont pas à être des valeurs numériques, comme c'est souvent le cas. PHP supporte 3 types de variables scalaires : les valeurs entières, les nombres à virgule flottante et les chaînes de caractères. (une variable scalaire est une variable que vous ne pouvez pas scinder en morceau, au contraire des tableaux par exemple). PHP supporte aussi deux types composés : les tableaux et les objets. Chacun de ces types de variables peuvent être affectés ou renvoyés par une fonction.

Les utilisateurs de PHP/FI 2 ne verront aucun changement. Malgré tout, PHP va plus loin dans la gestion des

expressions, comme le font d'autres langages. PHP est un langage orienté expression, dans le sens où presque tout est une expression. Considérons l'exemple dont nous avons déjà parlé, '\$a = 5'. Il est facile de voir que il y a deux valeurs qui entrent en jeu ici, la valeur numérique constante '5' et la valeur de la variable \$a qui est mis à jour à la valeur 5. Mais, la vérité est qu'il y a une autre valeur qui entre en jeu ici et c'est la valeur de l'assignement elle-même. L'assignement lui-même est assigné à une valeur, dans ce cas-là 5. En pratique, cela signifie que '\$a = 5' est une expression qui a pour valeur 5. Donc, en écrire '\$b = (\$a = 5)' revient à écrire '\$a = 5; \$b = 5;' (un point virgule marque la fin d'une instruction). Comme les assignements sont analysés de droite à gauche, vous pouvez aussi bien écrire '\$b = \$a = 5'.

Un autre bon exemple du langage orienté expression est la pré-incrémentation et la post-incrémentation, (ainsi que la décrément). Les utilisateurs de PHP/FI 2 et ceux de nombreux autres langages sont habitués à la notation "variable++" et "variable--". Ce sont les opérateurs d'incrément et de décrément. En PHP/FI 2, l'instruction '\$a++' n'a aucune valeur (c'est-à-dire que ce n'est pas une expression) et vous ne pouvez donc pas l'utiliser. PHP ajoute les possibilités d'incrément et de décrément comme c'est le cas dans le langage C. En PHP, comme en C, il y a deux types d'opérateurs d'incrément (pré-incrémentation et post-incrémentation). Les deux types d'opérateur d'incrément jouent le même rôle (c'est-à-dire qu'il incrémente la variable). La différence vient de la valeur de l'opérateur d'incrément. L'opérateur de pré-incrémentation, qui s'écrit '++\$variable', évalue la valeur incrémentée (PHP incrémente la variable avant de lire la valeur de cette variable, d'où le nom de 'pré-incrémentation'). L'opérateur de post-incrémentation, qui s'écrit '\$variable++', évalue la valeur de la variable avant de l'incrémenter. (PHP incrémente la variable après avoir lu sa valeur, d'où le nom de 'post-incrémentation'). Un type d'expression très commun est l'expression de comparaison. Ces expressions sont évaluées à 0 ou 1, autrement dit FALSE ou TRUE (respectivement). PHP supporte les opérateurs de comparaison > (plus grand que), => (plus grand ou égal), == (égal à), < (plus petit que), <= (plus petit ou égal). Ces expressions sont utilisées de manière courante dans les instructions conditionnelles, comme l'instruction if.

Pour le dernier exemple d'expression, nous allons parler des combinaisons d'opérateurs/assignement. Vous savez que si vous voulez incrémenter la variable \$a d'une unité, vous devez simplement écrire '\$a++'. Mais si vous voulez ajouter la valeur '3' à votre variable ? Vous pouvez écrire plusieurs fois '\$a++', mais ce n'est pas la meilleure des méthodes. Une pratique plus courante est d'écrire '\$a = \$a + 3'. L'expression '\$a + 3' correspond à la valeur \$a plus 3, et est de nouveau assignée à la variable \$a. Donc le résultat est l'incrément de 3 unités. En PHP, comme dans de nombreux autres langages comme le C, vous pouvez écrire cela de manière plus concise, manière qui avec le temps se révélera plus claire et plus rapide à comprendre. Ajouter 3 à la valeur de la variable \$a peut s'écrire '\$a += 3'. Cela signifie précisément : "on prend la valeur de la variable \$a, on ajoute la valeur 3 et on assigne cette valeur à la variable \$a". Et pour être plus concis et plus clair, cette expression est plus rapide. La valeur de l'expression '\$a += 3', comme l'assignement d'une valeur quelconque, est la valeur assignée. Il est à noter que ce n'est pas 3 mais la combinaison de la valeur de la variable \$a plus la valeur 3. (c'est la valeur qui est assignée à la variable \$a). N'importe quel opérateur binaire peu utiliser ce type d'assignement, par exemple '\$a -= 5' (soustraction de 5 de la valeur de la variable \$a), '\$b *= 7' (multiplication de la valeur de la variable \$b par 7).

Il y a une autre expression qui peut paraître complexe si vous ne l'avez pas vu dans d'autre langage, l'opérateur conditionnel ternaire:

```
$first ? $second : $third
```

Si la valeur de la première sous-expression est vraie, (différente de 0), alors la deuxième sous-expression est évaluée et constitue le résultat de l'expression conditionnelle. Sinon, c'est la troisième sous-expression qui est évaluée et qui constitue le résultat de l'expression.

Les exemples suivants devraient vous permettre de mieux comprendre la pré- et post- incrément et le concept des expressions en général:

```
function double($i) {
return $i*2;
```

```

}
$b = $a = 5;          /* assigne la valeur 5 aux variables $a et $b */
$c = $a++;           /* post-incrémentation de la variable $a et assignation de
la valeur à la variable $c */
$e = $d = ++$b;      /* Pré-incrémentation, et assignation de la valeur aux
variables $d et $e */
/* à ce niveau, les variables $d et $e sont égales à 6 */
$f = double($d++);  /* assignation du double de la valeur de $d à la variable $f ($f vaut 12),
puis incrémentation de la valeur de $d */
$g = double(++$e);  /* assigne deux fois la valeur de $e après
incrémentacion, 2*7 = 14 to $g */
$h = $g += 10;      /* Tout d'abord, $g est incrémentée de 10, et donc $g vaut 24.
Ensuite, la valeur de $g, (24) est assignée à la variable $h,
qui vaut donc elle aussi 24. */

```

Au début de ce chapitre, nous avons dit que nous allions décrire les différents types d'instructions, et donc, comme promis, nous allons voir que les expressions peuvent être des instructions. Mais, attention, toutes les expressions ne sont pas des instructions. Dans ce cas-là, une instruction est de la forme 'expr ';', c'est-à-dire, une expression suivie par un point virgule. L'expression '\$b = \$a = 5;', '\$a = 5' est une valide, mais ce n'est pas une instruction en elle-même. '\$b = \$a = 5' est une instruction valide.

La dernière chose qui mérite d'être mentionnée est la véritable valeur des expressions. Lorsque vous faites des tests sur une variable, dans une boucle conditionnelle par exemple, cela ne vous intéresse pas de savoir qu'elle est la valeur exacte de l'expression. Mais vous voulez seulement savoir si le résultat signifie TRUE ou FALSE (PHP n'a pas de type booléen). La véritable valeur d'une expression en PHP est calculée de la même manière qu'en Perl. Toute valeur numérique différente de 0 est considérée comme étant TRUE. Une chaîne de caractères vide et la chaîne de caractère 0 sont considérées comme FALSE. Toutes les autres valeurs sont vraies. Avec les types de variables non-scalaires, (les tableaux et les objets), s'ils ne contiennent aucun élément, renvoient FALSE, sinon, ils renvoient TRUE.

PHP propose une implémentation complète et détaillée des expressions. PHP documente toutes ses expressions dans le manuel que vous êtes en train de lire. Les exemples qui vont suivre devraient vous donner une bonne idée de ce qu'est une expression et comment construire vos propres expressions. Dans tout ce qui va suivre, nous écrirons *expr* pour indiquer tout expression PHP valide.

9.5 Fonctions

[\[Notes en ligne\]](#)

9.5.1 Les fonctions utilisateurs

[\[Notes en ligne\]](#) [\[Exemples\]](#)

une fonction peut être définie en utilisant la syntaxe suivante :

```

function foo ($arg_1, $arg_2, ..., $arg_n) {
echo "Exemple de fonction.\n";
return $retval;
}

```

Tout code PHP, correct syntaxiquement, peut apparaître dans une fonction et dans une définition de [9.6.1 class](#).

En PHP3, les fonctions doivent être définies avant qu'elles ne soient utilisées. Ce n'est plus le cas en PHP4. PHP ne supporte pas le surchargement de fonction, ni la destruction ou la redéfinition de fonctions déjà déclarées.

PHP3 ne supporte pas un nombre variable d'arguments (voir [9.5.2.2 Valeur par défaut des arguments](#) pour plus d'informations). PHP4 supporte les deux : voir [9.5.2.3 Variable-length argument lists](#) et les fonctions de références que sont [func_num_args\(\)](#), [func_get_arg\(\)](#), et [func_get_args\(\)](#) pour plus d'informations.

9.5.2 Les arguments de fonction

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Des informations peuvent être passées à une fonction en utilisant un tableau d'arguments, dont chaque élément est séparé par une virgule. Un élément peut être une variable ou une constante.

PHP supporte le passage d'arguments [9.5.2.2 Valeur par défaut des arguments](#) (méthode par défaut), par [9.5.2.1 Passage d'arguments par référence](#). Les listes variable d'arguments sont supportées par PHP4 et plus récent. Voir [9.5.2.3 Variable-length argument lists](#) et les fonctions utiles que sont [func_num_args\(\)](#), [func_get_arg\(\)](#), et [func_get_args\(\)](#). Fonctionnellement, on peut arriver au même résultat en passant un tableau comme argument :

```
function takes_array($input) {
echo "$input[0] + $input[1] = ", $input[0]+$input[1];
}
```

9.5.2.1 Passage d'arguments par référence

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Par défaut, les arguments sont passés à la fonction par valeur (donc vous pouvez changer la valeur d'un argument dans la fonction, cela ne change pas sa valeur à l'extérieur de la fonction). Si vous voulez que vos fonctions puisse changer la valeur des arguments, vous devez passer ces arguments par référence. Si vous voulez qu'un argument soit toujours passé par référence, vous pouvez ajouter un '&' devant l'argument dans la déclaration de la fonction :

```
function add_some_extra(&$string) {
    $string .= ', et un peu plus.';
}
$str = 'Ceci est une chaîne';
add_some_extra($str);
echo $str;    // affiche 'Ceci est une chaîne, et un peu plus.'
```

Si vous souhaitez passer une variable par référence à une fonction mais de manière ponctuelle, vous pouvez ajouter un '&' devant l'argument dans l'appel de la fonction:

```
function foo ($bar) {
    $bar .= ', et un peu plus.';
}
$str = 'Ceci est une chaîne';
foo ($str);
echo $str;    // affiche 'Ceci est une chaîne'
foo (&$str);
echo $str;    // affiche 'Ceci est une chaîne, et un peu plus.'
```

[9.5.2.2 Valeur par défaut des arguments](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Vous pouvez définir comme en C++ des valeurs par défaut pour les arguments de type scalaire :

```

function makecoffee ($type = "cappucino") {
return "Faire une tasse de $type.\n";
}
echo makecoffee ();
echo makecoffee ("thé");

```

La fonction ci-dessus affichera :

```

Faire une tasse de cappucino.
Faire une tasse de thé.

```

La valeur par défaut d'un argument doit obligatoirement être une constante, et ne peut être ni une variable ou ni un membre de classe.

Il est à noter que vous utilisez les arguments par défaut, la valeur par défaut doit se trouver du côté droit du signe '='; sinon, cela ne fonctionnera pas. Considérons le code suivant :

```

function makeyogurt ($type = "acidophilus", $flavour) {
return "Préparer un bol de $type $flavour.\n";
}
echo makeyogurt ("framboise"); // ne fonctionne pas comme voulu

```

L'affiche du code ci-dessus est le suivant :

```

Warning: Missing argument 2 in call to makeyogurt() in
/usr/local/etc/httpd/htdocs/php3test/functest.html on line 41
Préparer un bol de framboise.

```

Maintenant comparons l'exemple précédent avec l'exemple suivant :

```

function makeyogurt ($flavour, $type = "acidophilus") {
return "Préparer un bol de $type $flavour.\n";
}
echo makeyogurt ("framboise"); // fonctionne comme voulu

```

L'affichage de cette exemple est le suivant :

```

Préparer un bol de acidophilus framboise.

```

[9.5.2.3 Variable-length argument lists](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

PHP4 supporte le fonctions à nombre d'argument variable. C'es très simple à utiliser, avec les fonctions

[func_num_args\(\)](#), [func_get_arg\(\)](#), et [func_get_args\(\)](#).

Aucune syntaxe particulière n'est nécessaire, et la liste d'argument doit toujours être fournie explicitement avec la définition de la fonction, et se comportera comme normalement.

9.5.3 Les valeurs de retour

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Les valeurs sont renvoyées en utilisant une instruction de retour optionnelle. Tous types de variables peuvent être renvoyées, tableaux et objets compris.

```
function square ($num) {
return $num * $num;
}
echo square (4); // affiche '16'.
```

Vous ne pouvez pas renvoyer plusieurs valeurs en même temps, mais vous pouvez obtenir le même résultat en renvoyant un tableau.

```
function small_numbers() {
return array (0, 1, 2);
}
list ($zero, $one, $two) = small_numbers();
```

9.5.4 old function

[\[Notes en ligne\]](#) [\[Exemples\]](#)

L'instruction `old_function` vous permet de déclarer une fonction en utilisant une syntaxe du type PHP/FI2 (au détail près que vous devez remplacer l'instruction 'function' par 'old_function'.)

C'est une fonctionnalité obsolète et elle ne devrait être utilisée que dans le cadre de conversion de PHP/FI2 vers PHP3

Les fonctions déclarées comme `old_function` ne peuvent pas être appelée à partir du code interne du PHP.

Cela signifie, par exemple, que vous ne pouvez pas les utiliser avec des fonctions comme [usort\(\)](#), [array_walk\(\)](#), et [register_shutdown_function\(\)](#). Vous pouvez contourner ce problème en écrivant une fonction d'encapsulation qui appellera la fonction `old_function`.

9.5.5 Variable functions

[\[Notes en ligne\]](#) [\[Exemples\]](#)

PHP supporte le concept de fonctions variables. Cela signifie que si le nom d'une variable est entourée de parenthèses, PHP recherchera une fonction de même nom, et essaiera de l'exécuter. Cela peut servir, entre autre, lors de call back, de tables de fonctions...

Exemple de fonction variable

```
<?php
function foo() {
echo "dans foo(<br>\n";
```

```

}
function bar( $arg = '' ) {
echo "Dans bar(); l'argument était '$arg'.<br>\n";
}
$func = 'foo';
$func();
$func = 'bar';
$func( 'test' );
?>

```

9.6 Classes et objets

[\[Notes en ligne\]](#)

9.6.1 class

[\[Notes en ligne\]](#)

Une classe est une collection de variables et de fonctions qui concernent ces variables. Une classe est définie en utilisant la syntaxe suivante :

```

<?php
class Cart {
var $items; // Eléments de notre panier
    // Ajout de $num articles de type $artnr au panier
function add_item ( $artnr, $num ) {
    $this->items[ $artnr ] += $num;
}
    // Suppression de $num articles du type $artnr du panier
function remove_item ( $artnr, $num ) {
if ( $this->items[ $artnr ] > $num ) {
    $this->items[ $artnr ] -= $num;
return true;
} else {
return false;
}
}
}
?>

```

L'exemple ci-dessus définit la classe `Cart` qui est composée d'un tableau associatif contenant les articles du panier et de deux fonctions, une pour ajouter et une pour enlever des éléments au panier.

Les classes sont un type de variable. Pour créer une variable du type désiré, vous devez utiliser l'opérateur `new`.

```

$cart = new Cart;
$cart->add_item("10", 1);

```

L'instruction ci-dessus crée l'objet `$cart` de la classe `Cart`. La fonction `add_item()` est appelée afin d'ajouter l'article numéro 10 dans la panier.

Une classe peut être une extension d'une autre classe. Les classes "extended" ou "derived" héritent de toutes les variables et de toutes les fonctions de la classe père plus toutes les définitions que vous rajoutez à cette classe. Cela se fait avec le mot clef "extends". L'héritage multiple n'est pas supporté.

```
class Named_Cart extends Cart {
var $owner;
function set_owner ($name) {
    $this->owner = $name;
}
}
```

L'exemple ci-dessus définit la classe `Named_Cart` qui possède les même variables que la classe `Cart` et la variable `$owner` en plus, ainsi que la fonction `set_owner()`. Vous créez un panier nominatif de la même manière que précédemment, et vous pouvez alors affecter un nom au panier ou en connaître le nom. Vous pouvez de toutes les façons utiliser les même fonctions que sur un panier classique.

```
$ncart = new Named_Cart; // Création d'un panier nominatif
$ncart->set_owner ("kris"); // Affectation du nom du panier
print $ncart->owner; // Affichage du nom du panier
$ncart->add_item ("10", 1); // (héritage des fonctions de la classe père)
```

Dans les fonctions d'une classe, la variable `$this` est égale à l'objet de la classe. Vous pouvez utiliser la forme `"$this->quelquechose"` pour accéder aux fonctions ou aux variables de l'objet courant.

Le constructeur est la fonction qui est appelée automatiquement par la classe lorsque vous créez une nouvelle instance d'une classe. La fonction constructeur a le même nom que la classe.

```
class Auto_Cart extends Cart {
function Auto_Cart () {
    $this->add_item ("10", 1);
}
}
```

L'exemple ci-dessus définit la classe `Auto_Cart` qui hérite de la classe `Cart` et définit le constructeur de la classe. Ce dernier initialise le panier avec 1 article de type numéro 10 dès que l'instruction "new" est appelée. La fonction constructeur peut prendre ou non, des paramètres optionnels, ce qui la rend beaucoup plus pratique.

```
class Constructor_Cart extends Cart {
function Constructor_Cart ($item = "10", $num = 1) {
    $this->add_item ($item, $num);
}
}
// Place dans le caddie toujours la même chose...
$default_cart = new Constructor_Cart;
// Place dans le caddie des objets différents, comme dans la réalité
$different_cart = new Constructor_Cart ("20", 17);
```

Pour les classes qui utilisent l'héritage, le constructeur de la classe père n'est pas automatiquement appelé lorsque le constructeur de la classe dérivée est appelé.

[9.7 Les opérateurs](#)

[\[Notes en ligne\]](#)

9.7.1 Les opérateurs arithmétiques

[\[Notes en ligne\]](#)

Vous rappelez vous des opérations élémentaires apprises à l'école ?

exemple	nom	résultat
$\$a + \b	Addition	Somme de $\$a$ et $\$b$.
$\$a - \b	Soustraction	Différence de $\$a$ et $\$b$.
$\$a * \b	Multiplication	Produit de $\$a$ et $\$b$.
$\$a / \b	Division	Quotient de $\$a$ et $\$b$.
$\$a \% \b	Modulo	Reste de $\$a$ divisé par $\$b$.

9.7.2 Les opérateurs d'assignement

[\[Notes en ligne\]](#)

L'opérateur d'assignement le plus simple est le signe "=". Le premier réflexe est de penser que ce signe veut dire "égal à". Ce n'est pas le cas. Il signifie que l'opérande de gauche se voit affecter la valeur de l'expression qui est à droite du signe égal.

La valeur d'une expression d'assignement est la valeur assignée. Par exemple, la valeur de l'expression '\$a = 3' est la valeur 3. Cela permet de faire d'utiliser des astuces telles que : @exemple $\$a = (\$b = 4) + 5$; // \$a est maintenant égal à 9, et \$b vaut 4.

En plus du simple opérateur d'assignement, il existe des "opérateurs combinés" pour tous les opérateurs arithmétiques et pour les opérateurs sur les chaînes de caractères. Cela permet d'utiliser la valeur d'une variable dans une expression et d'affecter le résultat de cette expression à cette variable. Par exemple:

```
$a = 3;
$a += 5; // affecte la valeur 8 à la variable $a. (correspond à l'instruction '$a = $a + 5');
$b = "Hello ";
$b .= "There!"; // affecte la valeur "Bonjour ici!" à la variable $b (correspond à $b = $b."ici!");
```

On peut noter que l'assignement copie le contenu de la variable originale dans la nouvelle (assignement par valeur), ce qui fait que les changements de valeur d'une variable ne modifieront pas la valeur de l'autre. Cela peut se révéler important lors de la copie d'un grand tableau durant une boucle. PHP4 supporte aussi l'assignement par référence, en utilisant la syntaxe `$var = &$othervar`; , mais ce n'était pas possible en PHP3. 'L'assignement par référence' signifie que les deux variables contiennent les mêmes données, et que la modification de l'une affecte l'autre. D'un autre côté, la recopie est très rapide.

9.7.3 Bitwise Operators

[\[Notes en ligne\]](#)

Bitwise operators allow you to turn specific bits within an integer on or off.

exemple	nom	résultat
$\$a \& \b	ET (AND)	Les bits positionnés à 1 dans \$a ET dans \$b sont

		positionnés à 1.
$\$a \mid \b	OU (OR)	Bits that are set in either \$a or \$b are set.
$\$a \wedge \b	Xor	Les bits positionnés à 1 dans \$a OU dans \$b sont positionnés à 1.
$\sim \$a$	NON (Not)	Les bits qui sont positionnés à 1 dans \$a sont positionnés à 0, et vice versa.
$\$a \ll \b	Décalage à gauche	Décale les bits de \$a dans \$b par la gauche (chaque décalage équivaut à une multiplication par 2).
$\$a \gg \b	Décalage à droite	décalage des bits de \$a dans \$b par la droite (chaque décalage équivaut à une division par 2).

9.7.4 Opérateurs de comparaison

[\[Notes en ligne\]](#)

Les opérateurs de comparaison, comme le nom l'indique, vous permettent de comparer deux valeurs.

exemple	nom	résultat
$\$a == \b	Egal	Vrai si \$a est égal à \$b.
$\$a === \b	Identique	Vrai si \$a est égal à \$b et qu'ils sont de même type (PHP4 seulement).
$\$a != \b	Différent	Vrai si \$a est différent de \$b.
$\$a < \b	Plus petit que	Vrai si \$a est plus petit strictement que \$b.
$\$a > \b	Plus grand	Vrai si \$a est plus grand strictement que \$b.
$\$a <= \b	Inférieur ou égal	Vrai si \$a est plus petit ou égal à \$b.
$\$a >= \b	Supérieur ou égal	Vrai si \$a est plus grand ou égal à \$b.

Un autre opérateur conditionnelle est l'opérateur ternaire ("?:"), qui fonctionne comme en langage C.

```
(expr1) ? (expr2) : (expr3);
```

Cette expression renvoie la valeur de l'expression *expr2* si l'expression *expr1* est vraie, et l'expression *expr3* si l'expression *expr1* est fausse.

9.7.5 Opérateur de contrôle d'erreur

[\[Notes en ligne\]](#)

PHP supports one error control operator: the at sign (@). When prepended to an expression in PHP, any error messages that might be generated by that expression will be ignored.

If the [7.1.1.24 ini.track_errors](#) feature is enabled, any error message generated by the expression will be saved in the global variable `$php_errormsg`. This variable will be overwritten on each error, so check early if you want to use it.

```
<?php
/* Intentional SQL error (extra quote): */
$res = @mysql_query( "select name, code from 'namelist' ) or
die( "Query failed: error was '$php_errormsg'" );
?>
```

See also [error_reporting\(\)](#).

9.7.6 Opérateur d'exécutions

[\[Notes en ligne\]](#)

PHP supports one execution operator: backticks (```). Note that these are not single-quotes! PHP will attempt to execute the contents of the backticks as a shell command; the output will be returned (i.e., it won't simply be dumped to output; it can be assigned to a variable).

```
$output = `ls -al`;
echo "<pre>$output</pre>";
```

See also [system\(\)](#), [passthru\(\)](#), [exec\(\)](#), [popen\(\)](#), and [escapeshellcmd\(\)](#).

9.7.7 Incrementing/Decrementing Operators

[\[Notes en ligne\]](#)

PHP supports C-style pre- and post-increment and decrement operators.

example	name	effect
<code>++\$a</code>	Pre-increment	Increments <code>\$a</code> by one, then returns <code>\$a</code> .
<code>\$a++</code>	Post-increment	Returns <code>\$a</code> , then increments <code>\$a</code> by one.
<code>--\$a</code>	Pre-decrement	Decrements <code>\$a</code> by one, then returns <code>\$a</code> .
<code>\$a--</code>	Post-decrement	Returns <code>\$a</code> , then decrements <code>\$a</code> by one.

Here's a simple example script:

```
<?php
echo "<h3>Postincrement</h3>";
$a = 5;
echo "Should be 5: " . $a++ . "<br>\n";
echo "Should be 6: " . $a . "<br>\n";
echo "<h3>Preincrement</h3>";
$a = 5;
echo "Should be 6: " . ++$a . "<br>\n";
echo "Should be 6: " . $a . "<br>\n";
echo "<h3>Postdecrement</h3>";
$a = 5;
echo "Should be 5: " . $a-- . "<br>\n";
```

```

echo "Should be 4: " . $a . "<br>\n";
echo "<h3>Predecrement</h3>";
$a = 5;
echo "Should be 4: " . --$a . "<br>\n";
echo "Should be 4: " . $a . "<br>\n";
?>

```

9.7.8 Les opérateurs logiques

[\[Notes en ligne\]](#)

exemple	nom	résultat
\$a and \$b	ET (And)	Vrai si \$a ET \$b sont vrais.
\$a or \$b	OU (Or)	Vrai si \$a OU \$b est vrai
\$a xor \$b	XOR(Or)	Vrai si \$a OU \$b est vrai, mais pas les deux en même temps.
! \$a	NON (Not)	Vrai si \$a est faux.
\$a && \$b	ET (And)	Vrai si \$a ET \$b sont vrais.
\$a \$b	OU (Or)	Vrai si \$a OU \$b est vrai.

La raison pour laquelle il existe deux types de "ET" et de "OU" est qu'ils ont des priorités différentes. Voir le paragraphe [9.7.9 La précedence des opérateurs](#).

9.7.9 La précedence des opérateurs

[\[Notes en ligne\]](#)

La priorité des opérateurs spécifie l'ordre dans lequel les valeurs doivent être analysées. Par exemple, dans l'expression $1 + 5 * 3$, le résultat est 16 et non 18, car la multiplication ("*") à une priorité supérieure par rapport à l'addition ("+").

Le tableau suivant dresse une liste de la priorité des différents opérateurs dans un ordre croissant de priorité.

Associativité	Opérateurs
gauche	,
gauche	or
gauche	xor
gauche	and
droite	print
gauche	= += -- *= /= .= %= = ^= ~> <=> >>=
gauche	? :
gauche	
gauche	&&
gauche	

gauche	^
gauche	&
non-associative	== != ===
non-associative	< <= > >=
gauche	<< >>
gauche	+ - .
gauche	* / %
droite	! ~ ++ -- (int) (double) (string) (array) (object)
droite	[
non-associative	new

9.7.10 String Operators

[\[Notes en ligne\]](#)

Il y a deux opérateurs de chaînes. Le premier est l'opérateur de concaténation ('.'), qui retourne la concaténation de ses deux arguments. Le second est l'opérateur d'assignement concaténant ('.='). Reportez vous à [9.7.2 Les opérateurs d'assignement](#) pour plus de détails.

```
$a = "Bonjour ";
$b = $a . "Monde!"; // $b contient "Bonjour Monde!"
$a = "Bonjour ";
$a = $a . "Monde!"; // $a contient "Bonjour Monde!"
```

9.8 Types

[\[Notes en ligne\]](#)

PHP supporte les types suivants :

- [9.8.4 Les tableaux](#)
- [9.8.2 Les nombres à virgule flottante](#)
- [9.8.1 Entiers](#)
- [9.8.5 Les objets](#)
- [9.8.3 Les chaînes de caractères](#)

Habituellement, le type d'une variable n'est pas déclaré par le programmeur. Il est décidé au moment de l'exécution par le PHP, en fonction du contexte dans lequel la variable est utilisée.

Si vous voulez forcer une variable à être convertie en un certain type, vous devez transtyper ([9.8.6.1 Transtypage](#)) la variable ou utiliser la fonction [settype\(\)](#).

Il est à noter qu'une variable peut se comporter de manière différente suivant les situations, en fonction du type qui lui est affecté. Pour plus d'informations, voir le paragraphe [9.8.6 Définition du type](#).

9.8.1 Entiers

[\[Notes en ligne\]](#)

Il est possible de spécifier les nombres entiers (integers) de la manière suivante :

```
$a = 1234; # nombre entier en base 10
$a = -123; # nombre entier négatif
$a = 0123; # nombre entier en base 8, octale (équivalent à 83 en base 10)
$a = 0x12; # nombre entier en base 16, hexadécimale (équivalent à 18 en base 10)
```

9.8.2 Les nombres à virgule flottante

[\[Notes en ligne\]](#)

Les nombres à virgule flottante ("double") peuvent être spécifiés en utilisant la syntaxe suivante:

```
$a = 1.234;
$a = 1.2e3;
```

9.8.3 Les chaînes de caractères

[\[Notes en ligne\]](#)

Les chaînes de caractères peuvent être définies en utilisant deux types de délimiteurs.

Si la chaîne de caractères est délimitée par des guillemets doubles ("), les variables à l'intérieur de la chaîne seront évaluées, et remplacées par leur valeur. Comme en C ou en Perl, le caractère backslash (\) est utilisé pour protéger (échapper) un caractère spécial.

séquence	valeur
\n	Nouvelle ligne
\r	Retour à la ligne
\t	Tabulation horizontale
\\	Backslash
\\$	Caractère \$
\"	Guillemet double
\[0-7]{1,3}	Une séquence de caractère qui permet de rechercher un nombre en notation octale. @tab
\x[0-9A-Fa-f]{1,2}	Une séquence de caractère qui permet de rechercher un nombre en notation hexadécimale. @tab

Vous pouvez utiliser le caractère d'échappement backslash sur n'importe quel autre caractère, mais cela produira une alerte (si le niveau d'alerte maximal a été fixé).

Le deuxième moyen de délimiter une chaîne de caractère est d'utiliser les guillemets simples ('). Dans une telle chaîne de caractères, les variables *ne seront pas évaluées*, et le caractère backslash n'aura aucun effet (à deux exceptions près, pour "\\\" et "\", afin de pouvoir utiliser les caractères guillemets simples, et backslash dans la chaîne de caractères).

Another way to delimit strings is by using here doc syntax ("<<<"). One should provide an identifier after <<<, then the string, and then the same identifier to close the quotation. The closing identifier *must* begin in the first column of the line. ***Here doc string quoting example***

```
$str = <<<EOD
Example of string
spanning multiple lines
using heredoc syntax.
EOD;
```

Note : *Here doc support was added in PHP 4.*

Strings may be concatenated using the '.' (dot) operator. Note that the '+' (addition) operator will not work for this. Please see [9.7.10 String Operators](#) for more information.

Characters within strings may be accessed by treating the string as a numerically-indexed array of characters, using C-like syntax. See below for examples.

Some string examples

```
<?php
/* Assigning a string. */
$str = "This is a string";
/* Appending to it. */
$str = $str . " with some more text";
/* Another way to append, includes an escaped newline. */
$str .= " and a newline at the end.\n";
/* This string will end up being '<p>Number: 9</p>' */
$num = 9;
$str = "<p>Number: $num</p>";
/* This one will be '<p>Number: $num</p>' */
$num = 9;
$str = '<p>Number: $num</p>';
/* Get the first character of a string */
$str = 'This is a test.';
$first = $str[0];
/* Get the last character of a string. */
$str = 'This is still a test.';
$last = $str[strlen($str)-1];
?>
```

[9.8.3.1 Conversion de type](#)

[\[Notes en ligne\]](#)

Lorsqu'une chaîne de caractère est évaluée comme une valeur numérique, le résultat et le type de la variable sont déterminés comme suit.

La chaîne de caractères est de type "double" si elle contient un des caractère '.', 'e' ou 'E'. Sinon, elle est de type entier ("integer").

La valeur est définie par la première partie de la chaîne. Si la chaîne de caractères débute par une valeur numérique cette valeur sera celle utilisée. Sinon, la valeur sera égale à 0 (zéro).

Lorsque la première expression est une chaîne de caractères, le type de la variable dépend de la seconde expression.

```
$foo = 1 + "10.5";           // $foo est du type double (11.5)
$foo = 1 + "-1.3e3";        // $foo est du type double (-1299)
$foo = 1 + "bob-1.3e3";     // $foo est du type integer (1)
$foo = 1 + "bob3";         // $foo est du type integer (1)
$foo = 1 + "10 Small Pigs"; // $foo est du type integer (11)
$foo = 1 + "10 Little Piggies"; // $foo est du type integer (11)
$foo = "10.0 pigs " + 1;    // $foo est du type integer (11)
$foo = "10.0 pigs " + 1.0;  // $foo est du type double (11)
```

Pour plus d'informations sur les conversions de type, voir les pages de man à propos de la fonction `strtod(3)`. If you would like to test any of the examples in this section, you can cut and paste the examples and insert the following line to see for yourself what's going on:

```
echo "\$foo==\$foo; type is " . gettype( $foo ) . "<br>\n";
```

9.8.4 Les tableaux

[\[Notes en ligne\]](#)

Les tableaux ressemblent aux tables de hashage (tableaux associatifs) et aux tableaux indexés (vecteurs).

9.8.4.1 Tableaux à une dimension

[\[Notes en ligne\]](#)

PHP supporte les tableaux scalaires et les tableaux associatifs. En fait, il n'y a aucune différence entre les deux. Vous pouvez créer un tableau en utilisant les fonctions [list\(\)](#) ou [array\(\)](#), ou bien en affectant explicitement chacune des valeurs.

```
$a[0] = "abc";
$a[1] = "def";
$b["foo"] = 13;
```

Vous pouvez aussi créer un tableau en ajoutant simplement les valeurs à ce tableau.

```
$a[] = "hello"; // $a[2] == "hello"
$a[] = "world"; // $a[3] == "world"
```

Un tableau peut être trié en utilisant les fonctions [asort\(\)](#), [arsort\(\)](#), [ksort\(\)](#), [rsort\(\)](#), [sort\(\)](#), [uasort\(\)](#), [usort\(\)](#), ou [uksort\(\)](#) en fonction du type de classement que vous voulez.

Vous pouvez compter le nombre d'éléments qu'il y a dans un tableau en utilisant la fonction [count\(\)](#).

Vous pouvez vous déplacer à l'intérieur d'un tableau en utilisant les fonctions [next\(\)](#) et [prev\(\)](#). Un autre moyen de se déplacer dans un tableau est d'utiliser la fonction [each\(\)](#).

9.8.4.2 Tableaux à plusieurs dimensions

[\[Notes en ligne\]](#)

Les tableaux à plusieurs dimensions sont extrêmement simples. Pour chaque dimension du tableau, vous ajouter une nouvelle [clef] à la fin:

```

$a[1]          = $f;           # tableau à une dimension
$a["foo"]     = $f;
$a[1][0]      = $f;           # tableau à deux dimensions
$a["foo"][2]  = $f;           # (vous pouvez mélanger les indices associatifs et numériques)
$a[3]["bar"]  = $f;           # (vous pouvez mélanger les indices associatifs et numériques)
$a["foo"][4]["bar"][0] = $f; # tableau à quatre dimensions

```

En PHP3 il n'est pas possible de référencer un tableau à l'intérieur d'une chaîne. Par exemple, ceci ne fonctionne pas :

```

$a[3]['bar'] = 'Bob';
echo "Cela ne marche pas : $a[3][bar]";

```

En PHP3, l'exemple ci dessus va afficher : Cela ne marche pas : Array[bar]. L'opérateur de concaténation, peut être utilisé pour corriger cela :

```

$a[3]['bar'] = 'Bob';
echo "Cela ne marche pas : " . $a[3][bar];

```

En PHP4, cependant, le problème peut être contourné en entourant le tableau par des accolades :

```

$a[3]['bar'] = 'Bob';
echo "Cela marche : {$a[3][bar]}";

```

Vous pouvez remplir un tableau à plusieurs dimensions par de nombreux moyens mais la méthode la plus simple à comprendre est l'utilisation de la fonction [array\(\)](#). Les deux exemples suivants montre comment remplir un tableau à une dimension:

```

# Exemple 1:
$a["color"]    = "red";
$a["taste"]    = "sweet";
$a["shape"]    = "round";
$a["name"]     = "apple";
$a[3]          = 4;

# Exemple 2:
$a = array(
    "color" => "red",
    "taste" => "sweet",
    "shape" => "round",
    "name"  => "apple",
    3      => 4
);

```

La fonction [array\(\)](#) peut être emboîtée pour remplir un tableau à plusieurs dimensions :

<?

```

$a = array(
    "apple" => array(
        "color" => "red",
        "taste" => "sweet",
        "shape" => "round"
    ),
    "orange" => array(
        "color" => "orange",
        "taste" => "tart",
        "shape" => "round"
    ),
    "banana" => array(
        "color" => "yellow",
        "taste" => "paste-y",
        "shape" => "banana-shaped"
    )
);
echo $a["apple"]["taste"];    # will output "sweet"
?>

```

9.8.5 Les objets

[\[Notes en ligne\]](#)

9.8.5.1 Initialisation d'un objet

[\[Notes en ligne\]](#)

Pour initialiser un objet, vous devez utiliser la commande "new" afin de créer l'instance de l'objet.

```

class foo {
function do_foo () {
echo "Doing foo.";
}
}
$bar = new foo;
$bar->do_foo();

```

9.8.6 Définition du type

[\[Notes en ligne\]](#)

PHP ne nécessite pas de déclaration explicite du type d'une variable. Le type d'une variable est déterminé par le contexte d'utilisation. Par exemple, si vous assignez une chaîne de caractères à la variable *var*, var devient une chaîne de caractère. Si vous assignez un nombre entier à *var*, elle devient un entier.

Un exemple de convertisseur automatique de type est l'opérateur '+'. Si un des opérandes est de type double, alors tous les opérandes sont évalués comme des variables de type double et le résultat est de type double.

Sinon, tous les opérandes sont évalués comme des variables de type entier et le résultat sera du type entier. Il est à noter que cela NE CHANGE PAS le type des opérandes. Le seul changement est la manière dont les opérandes sont évaluées.

```

$foo = "0"; // $foo est une chaîne de caractères (ASCII 48)
$foo++;    // $foo est la chaîne de caractères "1" (ASCII 49)
$foo += 1; // $foo est maintenant du type entier (2)

```

```
$foo = $foo + 1.3; // $foo est maintenant du type double (3.3)
$foo = 5 + "10 Little Piggies"; // $foo est du type entier (15)
$foo = 5 + "10 Small Pigs"; // $foo est du type entier (15)
```

Si les deux derniers exemples vous semblent obscurs ou si vous voulez forcer une variable à être évaluée avec un certain type, reportez vous au paragraphe Conversion de type ci-dessous. Si vous voulez changer le type d'une variable, intéressez vous à la fonction [9.8.3.1 Conversion de type](#).

Si vous voulez forcer le type d'une variable, vous pouvez vous reporter à la section [9.8.6.1 Transtypage](#). Si vous voulez changer le type d'une variable, utilisez [settype\(\)](#).

If you would like to test any of the examples in this section, you can cut and paste the examples and insert the following line to see for yourself what's going on:

```
echo "\$foo==\$foo; type is " . gettype( $foo ) . "<br>\n";
```

Note : *The behaviour of an automatic conversion to array is currently undefined.*

```
$a = 1; // $a is an integer
$a[0] = "f"; // $a becomes an array, with $a[0] holding "f"
```

While the above example may seem like it should clearly result in \$a becoming an array, the first element of which is 'f', consider this:

```
$a = "1"; // $a is a string
$a[0] = "f"; // What about string offsets? What happens?
```

Since PHP supports indexing into strings via offsets using the same syntax as array indexing, the example above leads to a problem: should \$a become an array with its first element being "f", or should "f" become the first character of the string \$a?

For this reason, as of PHP 3.0.12 and PHP 4.0b3–RC4, the result of this automatic conversion is considered to be undefined. Fixes are, however, being discussed.

[9.8.6.1 Transtypage](#)

[\[Notes en ligne\]](#)

La conversion de type en PHP fonctionne de la même manière qu'en C: le nom du type désiré est écrit entre parenthèses devant la variable à transtyper ("cast").

```
$foo = 10; // $foo is an integer
$bar = (double) $foo; // $bar est un double
```

Les conversions autorisées sont:

- (int), (integer) – type entier
- (real), (double), (float) – type double
- (string) – ctype chaîne
- (array) – type tableau
- (object) – type objet

Il est à noter que les tabulations et les espaces sont autorisés à l'intérieur des parenthèses, donc les lignes suivantes sont équivalentes:

```
$foo = (int) $bar;
$foo = ( int ) $bar;
```

Le transtypage n'a pas toujours le résultat attendu. Par exemple :

Lorsque vous transtypez un scalaire ou une chaîne en tableau, la variable verra son contenu affecté au premier élément du tableau.

```
$var = 'ciao';
$arr = (array) $var;
echo $arr[0]; // outputs 'ciao'
```

Lorsque vous transtypez un scalaire ou une chaîne en objet, la valeur de la variable sera transformée en attribut de l'objet : l'attribut s'appellera 'scalar':

```
$var = 'ciao';
$obj = (object) $var;
echo $obj->scalar; // outputs 'ciao'
```

9.9 Les variables

[\[Notes en ligne\]](#)

9.9.1 Essentiel

[\[Notes en ligne\]](#)

En PHP, les variables sont représentées par un signe dollar suivi du nom de la variable. Le nom est sensible à la casse (ie : \$x != \$X).

```
$var = "Bob";
$Var = "Joe";
echo "$var, $Var"; // outputs "Bob, Joe"
```

En PHP3, les variables sont toujours assignées par valeur. C'est à dire, lorsque vous assignez une expression à une variable, la valeur de l'expression est copiée dans la variable. Cela signifie, par exemple, qu'après avoir assigné la valeur d'une variable à une autre, modifier une variable n'aura pas d'effet sur l'autre. Pour plus de détails sur ce genre d'assignement, reportez vous à [9.4 Les expressions](#).

PHP4 permet aussi d'assigner les valeurs aux variables *par référence*. Cela signifie que la nouvelle variable ne fait que référencer (en d'autres terme, "devient un alias de", ou encore "pointe sur") la variable originale. Les modifications de la nouvelle variable affecteront l'ancienne, et vice versa. Cela signifie aussi qu'aucune copie n'est faite : l'assignement est donc beaucoup plus rapide. Cela se fera notamment sentir dans des boucles, ou lors d'assignement de grands objets (tableaux).

Pour assigner par référence, ajoutez simplement un & (ET commercial) au début de la variable qui est assignée (la variable source). Dans l'exemple suivant, 'Mon nom est Pierre' s'affichera deux fois :

```
<?php
$foo = 'Pierre';           // Assigne la valeur 'Pierre' à $foo
$bar = &$foo;             // Référence $foo avec $bar.
$bar = "Mon nom est Pierre"; // Modifie $bar...
echo $foo;                // $foo est aussi modifiée
echo $bar;
?>
```

Une chose importante à noter est que seule les variables nommées peuvent être assignées par référence.

```
<?php
$foo = 25;
$bar = &$foo;           // Assignment valide .
$bar = &(24 * 7);      // Assignment Invalide : référence une expression sans nom
function test() {
return 25;
}
$bar = &test();        // Invalide.
?>
```

9.9.2 Variables prédéfinies

[\[Notes en ligne\]](#)

PHP fourni un grand nombre de variables prédéfinies. Cependant, beaucoup de ces variables ne peuvent pas être présentées ici, car elles dépendent du serveur sur lequel elles tournent, de la version du serveur, et de la configuration du serveur, ou encore d'autres facteurs.. Certaines de ces variables ne seront pas accessibles lorsque PHP fonctionne en exécutable.

Malgré ces données, voici une liste de variables prédéfinies, qui seront accessibles avec une installation ad hoc de PHP3, fonctionnant en module, sous [Apache 1.3.6](#).

Pour la liste complète des variables prédéfinies (et d'autres informations pratiques) reportez vous (et usez) de [phpinfo\(\)](#).

Note : Cette liste n'est pas exhaustive et ne le sera pas. C'est simplement un aperçu des variables prédéfinies qui peuvent être accessibles dans les scripts.

9.9.2.1 Variables Apache

[\[Notes en ligne\]](#)

Ces variables sont créées par le serveur [Apache](#). Si vous utilisez un autre serveur web, il n'est pas sur que celui ci vous fournira les mêmes variables. Il peut ne pas les fournir, en fournir d'autres. Cependant, un bon nombre de ces variables font partie de l'interface [CGI 1.1](#), et on peut s'attendre à les retrouver.

Notez que peu d'entre elles seront accessible lorsque PHP est appelé en ligne de commande, (et elles n'auront alors peut être pas de sens)

GATEWAY_INTERFACE

- Numéro de révision de l'interface CGI du serveur : i.e. 'CGI/1.1'.
SERVER_NAME
- Le nom du serveur hôte qui exécute le script suivant. Si le script est exécuté sur un hôte virtuel, ce sera la valeur définie pour cet hôte virtuel.
SERVER_SOFTWARE

- Chaîne d'identification du serveur, qui est données dans les entêtes lors de la réponse aux requêtes.
SERVER_PROTOCOL
- Nom et révision du protocole de communication : i.e. 'HTTP/1.0';
REQUEST_METHOD
- Méthode de requête utilisée pour accéder à la page; i.e. 'GET', 'HEAD', 'POST', 'PUT'.
QUERY_STRING
- La chaîne de requête, si elle existe, qui est utilisée pour accéder à la page.
DOCUMENT_ROOT
- La racine sous laquelle le script courant est exécuté, comme défini dans la configuration du serveur.
HTTP_ACCEPT
- Contenu de l'entête Accept: de la requête courant, si il y en a une.
HTTP_ACCEPT_CHARSET
- Contenu de l'entête Accept-Charset: de la requête courante, si il existe. Par exemple :
'iso-8859-1,*,utf-8'.
HTTP_ENCODING
- Contenu de l'entête Accept-Encoding: de la requête courante, si elle existe. Par exemple : 'gzip'.
HTTP_ACCEPT_LANGUAGE
- Contenu de l'entête Accept-Language: de la requête courante, si elle existe. Par exemple : 'en'.
HTTP_CONNECTION
- Contenu de l'entête Connection: de la requête courante, si elle existe. Par exemple : 'Keep-Alive'.
HTTP_HOST
- Contenu de l'entête Host: de la requête courante, si elle existe.
HTTP_REFERER
- L'adresse de la page (si elle existe) qui a conduit le client à la page courante. Cette valeur est affectée par le client, et tous les clients ne le font pas.
HTTP_USER_AGENT
- Contenu de l'entête User-Agent: de la requête courante, si elle existe. C'est une chaîne qui décrit le client HTML utilisé pour voir la page courante. Par exemple : Mozilla/4.5 [en] (X11; U; Linux 2.2.9 i586). Entre autres choses, vous pouvez utiliser cette valeur avec [get_browser\(\)](#) pour optimiser votre page en fonction des capacité du client.
REMOTE_ADDR
- L'adresse IP du client qui demande la page courante.
REMOTE_PORT
- Le port utilisé par la machine cliente pour communiquer avec le serveur web.
SCRIPT_FILENAME
- Le chemin absolu jusqu'au script courant.
SERVER_ADMIN
- La valeur donné à la directive SERVER_ADMIN (pour Apache), dans le fichier de configuration. Si le script est exécuté par un hôte virtuel, cela sera la valeur définie par l'hôte virtuel.
SERVER_PORT
- Le port de la machine serveur utilisé pour les communications. Par défaut, c'est '80'; en utilisant SSL, par exemple, il sera remplacé par le numéro de port HTTP sécurisé.
SERVER_SIGNATURE
- Chaîne contenant le numéro de version du serveur et le nom d'hôte virtuel, qui sont ajoutés aux pages générées par le serveur, si cette option est activée.
PATH_TRANSLATED
- Chemin dans le système de fichier (par le document root-) jusqu'au script courant, une fois que le serveur à fait une chemin traduction virtuel->réel.
SCRIPT_NAME
- Contient le nom du script courant. Cela sert lorsque les pages doivent s'appeler elles-mêmes.
REQUEST_URI

- L'URI qui a été fourni pour accéder à cette page. Par exemple : '/index.html'.

9.9.2.2 Variables d'environnement

[\[Notes en ligne\]](#)

Ces variables sont importées dans l'espace de nom global de PHP's, depuis l'environnement sous lequel PHP fonctionne. Beaucoup d'entre elles sont fournies par le shell qui exécute PHP et différents systèmes étant susceptibles de disposer de différents shells, une liste définitive est impossible à établir. Reportez vous à la documentation de votre shell, pour connaître la liste des variables d'environnement prédéfinies.

Les autres variables d'environnements inclues les variables CGI, placées ici, quelque fois la méthode d'exécution de PHP (CGI ou module).

9.9.2.3 Variables PHP

[\[Notes en ligne\]](#)

Ces variables sont créées par PHP lui-même.

argv

- Tableau des rguments passées au script. Lorsque le script est appelé en ligne de commande, cela donne accès aux arguments, comme en langage C. Lorsque le script est appelé avec la méthode GET, ce tableau contiendra la chaîne de requête.

argc

- Contient le nombre de paramètres de la ligne de commande passés au script (si il fonctionne en ligne de commande).

PHP_SELF

- Le nom du fichier du script en cour d'exécution, par rapport au document root. Si PHP fonctionne en ligne de commande, cette variable n'est pas disponible.

HTTP_COOKIE_VARS

- Un tableau associatif des variables passées au script courant via les HTTP cookies. Uniquement possible si le suivi des variable a été activé avec la directive générale [7.1.1.25 ini.track-vars](#) ou avec la directive locale `<?php_track_vars?>`.

HTTP_GET_VARS

- Un tableau associatif des variables passées au script courant via les HTTP GET. Uniquement possible si le suivi des variable a été activé avec la directive générale [7.1.1.25 ini.track-vars](#) ou avec la directive locale `<?php_track_vars?>`.

HTTP_POST_VARS

- Un tableau associatif des variables passées au script courant via les HTTP POST. Uniquement possible si le suivi des variable a été activé avec la directive générale [7.1.1.25 ini.track-vars](#) ou avec la directive locale `<?php_track_vars?>`.

9.9.3 Portée des variables

[\[Notes en ligne\]](#)

La portée d'une variable dépends du contexte dans lequel la variable est définie. Pour la plupart des variables, la portée concerne la totalité d'un script PHP. Mais, lorsque vous définissez une fonction, la portée d'une

variable définie dans cette fonction est locale à la fonction. Par exemple:

```
$a = 1;
include "b.inc";
```

Ici, la variable \$a sera accessible dans le script inclus `b.inc`. Cependant, dans les fonctions définies par l'utilisateur, une nouvelle définition de cette variable sera donnée, limitée à la fonction. Toute variable utilisée dans une fonction est par définition, locale. Par exemple :

```
$a = 1; /* portée global */
Function Test () {
echo $a; /* portée locale */
}
Test ();
```

Le script n'affichera rien à l'écran car la fonction [echo\(\)](#) utilise la variable local \$a, et celle-ci n'a pas été assignée préalablement dans la fonction. Vous pouvez noter que ce concept diffère un petit peu du langage C dans lequel une variable globale est automatiquement accessible dans les fonctions, à moins d'être redéfinie localement dans la fonction. Cela peut poser des problèmes si vous redéfinissez des variables globales localement. En PHP, une variable globale doit être déclarée à l'intérieure de chaque fonction afin de pouvoir être utilisée dans cette fonction. Par exemple:

```
$a = 1;
$b = 2;
Function Sum () {
global $a, $b;
    $b = $a + $b;
}
Sum ();
echo $b;
```

Le script ci-dessus va afficher la valeur "3". En déclarant global les variables \$a et \$b localement dans la fonction, toutes les références à ces variables concerneront les variables globales. Il n'y a aucune limite au nombre de variables globales qui peuvent être manipulées par une fonction.

Une deuxième méthode pour accéder aux variables globales est d'utiliser le tableau associatif prédéfini \$GLOBALS. Le précédent exemple peut être réécrit de la manière suivante:

```
$a = 1;
$b = 2;
Function Sum () {
    $GLOBALS["b"] = $GLOBALS["a"] + $GLOBALS["b"];
}
Sum ();
echo $b;
```

Le tableau \$GLOBALS est un tableau associatif avec le nom des variables globales comme clef et les valeurs des éléments du tableau comme valeur des variables.

Une autre caractéristique importante de la portée des variables est la notion de variable *static*. Une variable statique a une portée locale uniquement mais elle ne perd pas sa valeur lorsque le script appelle la fonction. Prenons l'exemple suivant:

```
Function Test () {
    $a = 0;
echo $a;
    $a++;
}
```

Cette fonction est un peu inutile car à chaque fois qu'elle est appelée, elle initialise \$a à 0 et affiche "0". L'incréméntation de la variable (\$a++) ne sert pas à grand chose car dès que la fonction est terminée la variable disparaît. Pour faire une fonction de comptage utile, c'est-à-dire qui ne perdra pas la trace du compteur, la variable \$a est déclarée comme une variable static:

```
Function Test () {
static $a = 0;
echo $a;
    $a++;
}
```

Maintenant, à chaque fois que la fonction Test() est appelée, elle affichera une valeur de \$a incrémentée de 1. Les variables static sont essentielles lorsque vous faites des appels récursifs à une fonction. Une fonction récursive est une fonction qui s'appelle elle-même. Il faut faire attention lorsque vous écrivez une fonction récursive car il est facile de faire une boucle infinie. Vous devez vérifier que vous avez bien une condition qui permet de terminer votre récursivité. La fonction suivante compte récursivement jusqu'à 10:

```
Function Test () {
static $count = 0;
    $count++;
echo $count;
if ($count < 10) {
Test ();
}
    $count--;
}
```

9.9.4 Les variables dynamiques

[\[Notes en ligne\]](#)

Il est pratique d'avoir parfois des noms de variables qui sont variables. C'est-à-dire un nom de variable qui affecté et utilisé dynamiquement. Une variable classique est affecté avec l'instruction suivante:

```
$a = "bonjour";
```

Une variable dynamique prend la valeur d'une variable et l'utilise comme nom d'une autre variable. Dans l'exemple ci-dessous, *hello*, peut être utilisé comme le nom d'une variable en utilisant le "\$\$" précédent la variable. c'est-à-dire

```
$$a = "world";
```

A ce niveau, deux variables ont été définies et stockées dans l'arbre des symboles PHP: \$a avec comme valeur "hello" et \$hello avec comme valeur "world". Alors, l'instruction

```
echo "$a ${$a}";
```

produira le même affichage que :

```
echo "$a $hello";
```

c'est-à-dire : *hello world*.

Afin de pouvoir utiliser les variables dynamiques avec les tableaux, vous avez à résoudre un problème ambigu. Si vous écrivez \$\$a[1], le parseur a besoin de savoir si vous parlez de la variable qui a pour nom \$a[1] ou bien si vous voulez l'index [1] de la variable \$\$a. La syntaxe pour résoudre cette ambiguïté est la

suivante: `$$a[1]` pour le premier cas, et `$$a}[1]` pour le deuxième.

[9.9.5 Variables externes à PHP](#)

[\[Notes en ligne\]](#)

[9.9.5.1 Formulaires HTML \(GET et POST\)](#)

[\[Notes en ligne\]](#)

Lorsqu'un formulaire est envoyé à un script PHP, toutes les variables du formulaire seront automatiquement disponibles dans le script. Par exemple, considérons le formulaire suivant:

Exemple avec un formulaire simple

```
<form action="foo.php3" method="post">
Name: <input type="text" name="name"><br>
      <input type="submit">
</form>
```

Lorsque ce formulaire est envoyé, le PHP va créer la variable `$name`, qui contiendra la valeur que vous avez entré dans le champs *Name*: du formulaire.

Le PHP permet aussi l'utilisation des tableaux dans le contexte de formulaire, mais seulement des tableaux à une seule dimension. Comme cela, vous pouvez rassembler des variables ou utiliser cette fonctionnalité pour récupérer les valeurs d'un choix multiple :

Variables complexes de formulaire

```
<form action="array.php" method="post">
Name: <input type="text" name="personal[name]"><br>
Email: <input type="text" name="personal[email]"><br>
Beer: <br>
      <select multiple name="beer[]">
        <option value="warthog">Warthog
        <option value="guinness">Guinness
        <option value="stuttgarter">Stuttgarter Schwabenbräu
      </select>
      <input type="submit">
</form>
```

Si l'option "track_vars" est activée, soit par l'option de compilation [7.1.1.25 ini.track-args](#), soit par la directive de configuration `<?php_track_vars?>`, les variables transmises par les méthodes POST et GET pourront aussi être trouvées dans le tableau associatif global `$HTTP_POST_VARS` ou `$HTTP_GET_VARS` suivant la méthode utilisée.

[9.9.5.1. Bouton "submit" sous forme d'image](#)

[\[Notes en ligne\]](#)

Lorsque vous envoyez le résultat d'un formulaire, vous pouvez utiliser une image au lieu du bouton "submit" standard en utilisant un tag :

```
<input type="image" src="image.gif" name="sub">
```

Lorsqu'un utilisateur clique sur l'image, le formulaire sera transmis au serveur avec deux variables de plus, `sub_x` et `sub_y`. Ces deux variables contiennent les coordonnées de l'endroit où l'utilisateur a cliqué. Les utilisateurs expérimentés remarqueront que les noms de variables sont transmis avec une virgule à la place du caractère "_", mais le PHP fait la conversion automatiquement.

[9.9.5.2 HTTP Cookies](#)

[\[Notes en ligne\]](#)

Le PHP supporte les cookies HTTP de manière totalement transparente, comme défini dans les [Netscape's Spec](#). Les cookies sont un mécanisme permettant de stocker des données sur la machine cliente à des fins d'authentification de l'utilisateur. Vous pouvez établir un cookie grâce à la fonction [setcookie\(\)](#). Les cookies font partie intégrante du "header" HTTP, et donc la fonction [setcookie\(\)](#) doit être appelé avant que le moindre affichage ne soit envoyé au navigateur. C'est la même restriction que pour la fonction [header\(\)](#). Tout cookie envoyé depuis le client sur le serveur sera automatiquement stocké sous forme de variable, comme pour la méthode POST ou GET.

Si vous souhaitez assigner plusieurs valeurs à un seul cookie, il vous faut ajouter les caractères `[]` au nom de votre cookie. Par exemple :

```
SetCookie ("MyCookie[]", "Testing", time()+3600);
```

Il est à noter qu'un cookie remplace le cookie précédent par un cookie de même nom tant que le "path" ou le domaine sont identiques. Donc, pour un "shopping cart", vous devez implémenter un compteur et l'incrémenter au fur et à mesure. C'est-à-dire:

Exemple SetCookie

```
$Count++;
SetCookie ("Count", $Count, time()+3600);
SetCookie ("Cart[$Count]", $item, time()+3600);
```

[9.9.5.3 Variables d'environnement](#)

[\[Notes en ligne\]](#)

Le PHP fait en sorte que les variables d'environnement soient accessibles directement comme des variables PHP normales.

```
echo $HOME; /* Affiche la valeur de la variable d'environnement HOME, si celle-ci est affectée.
```

Même si le PHP crée les variables lors de l'utilisation des méthodes GET, POST et cookie, il est de temps en temps préférable de transmettre explicitement la valeur de la variable afin d'être sûr de la valeur. La fonction [getenv\(\)](#) peut être utilisé pour récupérer la valeur des variables d'environnement. Vous pouvez aussi affecter une variable d'environnement grâce à la fonction [putenv\(\)](#).

[9.9.5.4 Cas des points dans les noms de variables](#)

[\[Notes en ligne\]](#)

Typiquement, PHP ne modifie pas les noms des variables lorsqu'elles sont passées à un script. Cependant, il faut noter que les points (.) ne sont pas autorisés dans les noms de variables PHP. Pour cette raison, jetez un oeil sur :

```
$varname.ext; /* invalid variable name */
```

Dans ce cas, l'analyseur croit voir la variable nommée \$varname, suivie par l'opérateur de concaténation, et suivi encore par la chaîne non-guillemetée (une chaîne sans guillemets, et qui n'a pas de signification particulière). Visiblement, ce n'est pas ce qu'on attendait...

Pour cette raison, il est important de noter que PHP remplacera automatiquement les points des noms de variables entrantes par des soulignés (underscore).

[9.9.5.5 Détermination du type des variables](#)

[\[Notes en ligne\]](#)

Parce que le PHP détermine le type des variables et les converties (généralement) comme il faut, ce n'est pas toujours le type de variable que vous souhaitez. PHP inclus des fonctions permettant de déterminer le type d'une variable. Les fonctions [gettype\(\)](#), [is_long\(\)](#), [is_double\(\)](#), [is_string\(\)](#), [is_array\(\)](#), et [is_object\(\)](#).

[9.10 References Explained](#)

[\[Notes en ligne\]](#)

[9.10.1 What are References](#)

[\[Notes en ligne\]](#)

References in PHP are means to call same variable content with different names. They are not like C pointers, they are symbol table aliases. Note that in PHP, variable names and variable content are different, so same content can have different names. The most close analogy is Unix filenames and files – variable names are directory entries, while variable contents is the file itself. References can be thought of as hardlinking in Unix filesystem.

[9.10.2 What do References](#)

[\[Notes en ligne\]](#)

PHP references allow you to make two variables to refer to the same content. Meaning, when you do:

```
$a =& $b
```

it means that *\$a* and *\$b* point to the same variable. Note : *\$a* and *\$b* are completely equal here, that's not *\$a* is pointing to *\$b* or vice versa, that's *\$a* and *\$b* pointing to the same place.

The second thing references do is to pass variables by-reference. This is done by making local function variable and caller variable to be reference to the same content. Example:

```
function foo (&$var) {
    $var++;
}
$a=5;
foo ($a);
```

will make *\$a* to be 6. This happens because in the function *foo* the variable *\$var* refers to the same content as *\$a*.

The third thing reference can do is [9.10.4 Returning References](#).

[9.10.3 What aren't References](#)

[\[Notes en ligne\]](#)

As said above, references aren't pointers. That means, the following construct won't do what you expect:

```
function foo (&$var) {
    $var =& $GLOBALS["baz"];
}
foo($bar);
```

What will happen that *\$var* in *foo* will be bound with *\$bar* in caller, but then it will be re-bound with *\$GLOBALS["baz"]*. There's no way to bind *\$bar* in the caller to something else using reference mechanism, since *\$bar* is not available in the function *foo* (it is represented by *\$var*, but *\$var* has only variable contents and not name-to-value binding in the calling symbol table).

[9.10.4 Returning References](#)

[\[Notes en ligne\]](#)

Returning by-reference it is useful when you want to use function to find variable which should be bound to. When returning references, use this syntax:

```
function &find_var ($param) {
    ...code...
    return $found_var;
}
$foo =& find_var ($bar);
$foo->x = 2;
```

In this example, property of the object returned by the *find_var* function would be set, not of the copy, as it would be without using reference syntax.

Note : *Unlike parameter passing, here you have to use & in both places – to indicate that you return by-reference, not a copy as usual, and to indicate than reference binding and not usual assignment should be done for \$foo.*

[9.10.5 Unsetting References](#)

[\[Notes en ligne\]](#)

When you unset the reference, you just break the binding between variable name and variable content. This does not mean that variable content will be destroyed. For example:

```
$a = 1;
$b =& $a;
unset ($a);
```

won't unset *\$b*, just *\$a*.

Again, it might be useful to think about this as analogous to Unix `unlink` call.

9.10.6 Spotting the Reference

[\[Notes en ligne\]](#)

Many syntax constructs in PHP are implemented via referencing mechanisms, so everything told above about reference binding also apply to these constructs. Some constructs, like passing and returning by-reference, are mentioned above. Other constructs that use references are:

9.10.6.1 global References

[\[Notes en ligne\]](#)

When you declare variable as `global $var` you are in fact creating reference to a global variable. That means, this is the same as:

```
$var =& $GLOBALS["var"];
```

That means, for example, that unsetting *\$var* won't unset global variable.

9.10.6.2 \$this

[\[Notes en ligne\]](#)

In an object method, *\$this* is always reference to the caller object.

10 Fonctions

[\[Notes en ligne\]](#) [\[Exemples\]](#)

10.1 Apache

[\[Notes en ligne\]](#)

10.1.1 apache lookup uri

[\[Notes en ligne\]](#) [\[Exemples\]](#)

class [apache_lookup_uri](#)(string *filename*)

Effectue une requête partielle pour l'URI spécifiée. Cette requête permet de récupérer toutes les informations importantes à propos de la ressource concernée. Les propriétés de la classe renvoyée sont les suivantes :

- status
- the_request
- status_line
- method
- content_type
- handler
- uri
- filename
- path_info
- args
- boundary
- no_cache
- no_local_copy
- allowed
- send_bodyct
- bytes_sent
- byterange
- clength
- unparsed_uri
- mtime
- request_time

Note : *Note: apache_lookup_uri ne fonctionne que lorsque le PHP est installé sous la forme d'un module Apache.*

10.1.2 apache note

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [apache_note](#)(string *note_name*, string *note_value*)

[apache_note\(\)](#) est une fonction spécifique au serveur Apache. Cette fonction affecte ou renvoie la valeur de la variable contenue dans la table notes d'Apache. Si la fonction est appelée avec un argument, elle renvoie la

valeur courante de la variable `note_name`. Si la fonction est appelée avec deux arguments, la variable `note_name` la valeur `note_value` et la fonction retournera la valeur précédente de la variable `note_name`.

[10.1.3 getallheaders](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [getallheaders](#),

Cette fonction renvoie un tableau associatif de tous les entêtes HTTP correspondants à la requête courante.

Note : Note: Vous pouvez récupérer la valeur d'une variable d'une CGI en la lisant à partir des variables d'environnement, ce qui fonctionne aussi bien dans le cas d'une installation en module ou en CGI. Utilisez la fonction [phpinfo\(\)](#) pour avoir une liste de toutes les variables d'environnement disponibles.

Exemple d'utilisation de la fonction [getallheaders\(\)](#)

```
$headers = getallheaders();
while (list($header, $value) = each($headers)) {
echo "$header: $value<br>\n";
}
```

Cette exemple est un exemple d'affichage de toutes les entêtes de la requête courante. *Note : Note: La fonction [getallheaders\(\)](#) ne fonctionne que si PHP est installé comme module Apache.*

[10.1.4 virtual](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [virtual](#)(string *filename*)

[virtual\(\)](#) est une fonction spécifique au serveur Apache. Elle est équivalente à la directive "`<!--#include virtual...-->`" lorsque vous utilisez le module include d'Apache. Cette fonction effectue une sous-requête Apache. C'est très utile lorsque vous utilisez des scripts CGI, des fichiers .shtml ou n'importe quel type de fichier qui doit être analysé par le serveur Apache. Il est à noter que lors de l'utilisation avec des scripts CGI, ces derniers doivent générer une entête valide, c'est-à-dire, au minimum une entête "Content-Type". Pour les fichiers PHP, il est conseillé d'utiliser les fonctions [include\(\)](#) et [require\(\)](#). [virtual\(\)](#) ne peut pas être utilisé pour inclure un fichier qui est lui même un fichier PHP.

[10.1.5 ascii2ebcdic](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ascii2ebcdic](#)(string *ascii_str*)

[ascii2ebcdic\(\)](#) est une fonction spécifique à Apache, qui n'est disponible que sur les OS qui gère le format EBCDIC (OS/390, BS2000). Elle traduit la chaîne ASCII *ascii_str* en sont équivalent EBCDIC (avec protection des données binaires) et retourne le résultat.

Voir aussi la fonction complémentaire : [ebcdic2ascii\(\)](#)

10.1.6 ebcidic2ascii

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ebcidic2ascii](#)(string *ebcidic_str*)

[ebcidic2ascii\(\)](#) est une fonction spécifique à Apache, qui n'est disponible que sur les OS qui gère le format EBCDIC (OS/390, BS2000). Elle traduit la chaîne EBCDIC *ebcidic_str* en son équivalent ASCII (avec protection des données binaires) et retourne le résultat.

Voir aussi la fonction complémentaire : [ascii2ebcidic\(\)](#)

10.2 Tableaux

[\[Notes en ligne\]](#)

Ces fonctions vous permettent de manipuler et de traiter les tableaux de nombreuses façons. Les tableaux sont très efficaces dès qu'il s'agit de stocker, gérer et traiter des données en groupe.

Les tableaux simples et multi-dimensionnels sont supportés et peuvent être créés par l'utilisateur, ou par une fonction. Il y a des fonctions spécifiques qui remplissent des tableaux à partir de résultats de requêtes, et de nombreuses fonctions retournent un tableau.

Voir aussi [is_array\(\)](#), [explode\(\)](#), [implode\(\)](#), [split\(\)](#) et [join\(\)](#).

10.2.1 array

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array](#)(mixed ...)

Retourne un tableau, créé à partir des paramètres fournis. Les paramètres peuvent avoir un index, fournis sous la forme clé => valeur.

Note : [array\(\)](#) n'est pas une fonction standard, elle existe simplement pour représenter littéralement des tableaux.

Les exemples suivants montrent la construction de tableaux bi-dimensionnels, l'assignation de clés pour les tableaux associatifs, et comment écarter certains intervalles d'indices numériques. **Exemples avec [array\(\)](#)**

```
$fruits = array (
    "fruits" => array("a"=>"orange", "b"=>"banane", "c"=>"pomme"),
    "nombres" => array(1, 2, 3, 4, 5, 6),
    "trous" => array("premier", 5 => "deuxième", "troisième")
);
```

Voir aussi : [list\(\)](#).

10.2.2 array_count_values

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_count_values](#)(array *input*)

[array_count_values\(\)](#) retourne un tableau contenant les valeurs du tableau *input* comme clés et leurs fréquence. *input* comme valeur.

Exemple avec [array_count_values\(\)](#)

```
$array = array(1, "bonjour", 1, "monde", "bonjour");
array_count_values($array); // retourne array(1=>2, "bonjour"=>2, "monde"=>1)
```

Note : Cette fonction a été ajoutée dans PHP 4.0.

10.2.3 array_diff

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_diff](#) (array *array1*, array *array2*, array ...)

[array_diff\(\)](#) retourne un tableau qui contient toutes les valeurs du tableau *array1* qui sont absentes de tous les autres arguments. Notez que les clés sont préservées.

Exemple avec [array_diff\(\)](#)

```
$array1 = array ("a" => "vert", "rouge", "bleu");
$array2 = array ("b" => "vert", "jaune", "rouge");
$result = array_diff ($array1, $array2);
```

\$result contient array ("bleu");

Voir aussi [array_intersect\(\)](#).

10.2.4 array_flip

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_flip](#) (array *trans*)

[array_flip\(\)](#) retourne un tableau dont les clés sont les valeurs du précédent tableau, et les valeurs sont les clés.

Exemple avec [array_flip\(\)](#)

```
$trans = array_flip ($strans);
$original = strtr ($str, $trans);
```

Note : Cette fonction a été ajoutée dans PHP 4.0.

10.2.5 array_intersect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_intersect](#) (array *array1*, array *array2*, array ...)

[array_intersect\(\)](#) retourne un tableau contenant toutes les valeurs de *array1* qui sont présentes dans tous les autres arguments. Notez que les clés sont préservées.

Exemple avec [array_intersect\(\)](#)

```
$array1 = array ("a" => "vert", "rouge", "bleu");
$array2 = array ("b" => "vert", "jaune", "rouge");
$result = array_intersect ($array1, $array2);
```

\$result contient array ("a" => "vert", "rouge");

Voir aussi [array_diff\(\)](#).

10.2.6 array_keys

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_keys](#) (array *input*, mixed *search_value*)

[array_keys\(\)](#) retourne les clés numériques et littérales du tableau *input*.

Si l'option *search_value* est spécifiée, seule les clés ayant cette valeur seront retournées. Sinon, toutes les clés de *input* sont retournées.

Exemple avec [array_keys\(\)](#)

```
$array = array(0 => 100, "couleur" => "rouge");
array_keys ($array); // retourne array (0, "couleur")
$array = array(1, 100, 2, 100);
array_keys ($array, 100); // retourne array (0, 2)
```

Note : Cette fonction a été ajoutée dans PHP 4. Ci dessous, voici une implémentation qui fonctionnera sous PHP3: implémentation de [array_keys\(\)](#) pour les utilisateurs de php 3 implémentation de [array_keys\(\)](#) pour les utilisateurs de php 3

```
function array_keys ($arr, $term="") {
    $t = array();
    while (list($k,$v) = each($arr)) {
        if ($term && $v != $term)
            continue;
        $t[] = $k;
    }
    return $t;
}
```

} Voir aussi [array_values\(\)](#).

10.2.7 array_merge

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_merge](#) (array *array1*, array *array2*, ...)

[array_merge\(\)](#) rassemble les éléments de plusieurs tableaux ensemble, en ajoutant les valeurs de l'un à la fin de l'autre. Le résultat est un tableau.

Si les tableaux ont des clés en commun, la dernière valeur rencontrée écrasera l'ancienne. Pour les valeurs numériques, cela n'arrive pas, car alors, les valeurs sont ajoutées en fin de tableau.

Exemple avec [array_merge\(\)](#)

```
$array1 = array ("couleur" => "rouge", 2, 4);
$array2 = array ("a", "b", "couleur" => "vert", "forme" => "trapézoïde");
array_merge ($array1, $array2);
```

Le résultat sera array("couleur" => "vert", 2, 4, "a", "b", "forme" => "trapézoïde").

Note : Cette fonction a été ajoutée dans PHP 4.0.

10.2.8 array_merge_recursive

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_merge_recursive](#) (array *array1*, array *array2*, array ...)

[array_merge_recursive\(\)](#) rassemble tous les éléments de plusieurs tableaux ensemble, en ajoutant les éléments de l'un à la suite des éléments du précédent. Cette fonction retourne le tableau résultant.

Si les tableaux passés en arguments ont les mêmes clés (chaînes de caractères), les valeurs sont alors rassemblées dans un tableau, de manière récursive, de façon à ce que, si l'une de ces valeurs est un tableau elle-même, la fonction la rassemblera avec les valeurs de l'entrée courante. Cependant, si deux tableaux ont la même clé numérique, la dernière valeur n'écrasera pas la précédente, mais sera ajouté à la fin du tableau.

Exemple avec [array_merge_recursive\(\)](#)

```
$ar1 = array ("couleur" => array ("favorie" => "red"), 5);
$ar2 = array (10, "couleur" => array ("favorie" => "vert", "rouge"));
$result = array_merge_recursive ($ar1, $ar2);
```

Le résultat sera array ("couleur" => array ("favorie" => array ("rouge", "vert"), "bleu"), 5, 10).

Voir aussi [array_merge\(\)](#).

10.2.9 array_multisort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [array_multisort](#) (array *ar1*, mixed *arg* , mixed ... , array ...)

[array_multisort\(\)](#) sert à trier simultanément plusieurs tableaux, ou bien à trier un tableau multi-dimensionnel, suivant l'une ou l'autre de ses dimensions. Les clés sont préservées.

Les tableaux passés en arguments sont traités comme les colonnes d'une table, triées par lignes (un peu comme la clause SQL ORDER BY). Le premier tableau est la clé primaire de tri. Les valeurs du premier tableau qui sont égales, sont triées grâce au tableau suivant, et ainsi de suite...

La structure des arguments de cette fonction est un peu inhabituelle, mais elle est plus souple. Le premier argument DOIT être un tableau, mais les arguments suivants peuvent être des tableaux ou une ou deux options de tri, prises dans les valeurs suivantes :

Options de tri :

- SORT_ASC – Tri en ordre ascendant
- SORT_DESC – Tri en ordre descendant

Sorting type flags:

- SORT_REGULAR – Comparaison normale des valeurs
- SORT_NUMERIC – Comparaison numérique des valeurs
- SORT_STRING – Comparaison alphabétique des valeurs

Une seule option de tri de chaque type peut être appliquée après un tableau. Une option ne s'applique qu'au tableau précédent. Tous les autres sont mis par défaut à SORT_ASC et SORT_REGULAR.

Retourne true en cas de succès, false sinon.

Trier plusieurs tableaux

```
$ar1 = array ("10", 100, 100, "a");
$ar2 = array (1, 3, "2", 1);
array_multisort ($ar1, $ar2);
```

Dans cet exemple, Dans cet exemple, après le tri, le premier tableau contient 10, "a", 100, 100; Le deuxième tableau contient 1, 1, 2, "3". Les entrées du second tableau correspondent aux valeurs jumelles du premier tableau (100 et 100), sont aussi triées.

Classer un tableau multidimensionnel

```
$ar = array (array ("10", 100, 100, "a"), array (1, 3, "2", 1));
array_multisort ($ar[0], SORT_ASC, SORT_STRING,
                $ar[1], SORT_NUMERIC, SORT_DESC);
```

Dans cet exemple, après le tri, le premier tableau contient 10, 100, 100, "a" (tri alphabétique, ordre croissant); Le deuxième tableau contient 1, 3, "2", 1 (tri numérique, ordre décroissant).

10.2.10 array_pad

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_pad](#) (array *input*, int *pad_size*, mixed *pad_value*)

[array_pad\(\)](#) retourne une copie du tableau *input* complété jusqu'à la taille de *pad_size* avec la valeur *pad_value*. si *pad_size* est positif alors le tableau est complété à droite, si il est négatif, il est complété à gauche. Si la valeur absolue de *pad_size* est plus petite que la taille du tableau *input* alors le tableau n'est pas complété.

Exemple avec [array_pad\(\)](#)

```
$input = array (12, 10, 9);
$result = array_pad ($input, 5, 0);
// Le résultat est array (12, 10, 9, 0, 0)
$result = array_pad ($input, -7, -1);
// Le résultat est array (-1, -1, -1, -1, 12, 10, 9)
$result = array_pad ($input, 2, "noop");
// pas complété
```

[10.2.11 array_pop](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [array_pop](#)(array *array*)

[array_pop\(\)](#) dépile et retourne le dernier élément du tableau *array*, le raccourcissant d'un élément.

Exemple avec [array_pop\(\)](#)

```
$stack = array ("orange", "pomme", "framboise");
$fruit = array_pop ($stack);
```

Après ceci, \$stack n'a plus que 2 éléments: "orange" et "pomme", tandis que \$fruit contient "framboise".

Voir aussi [array_push\(\)](#), [array_shift\(\)](#), et [array_unshift\(\)](#). Note : *Cette fonction a été ajoutée dans PHP 4.0.*

[10.2.12 array_push](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [array_push](#)(array *array*, mixed *var*, ...)

[array_push\(\)](#) considère *array* comme une pile, et empile les variables passées en paramètres à la fin de *array*. La longueur du tableau *array* augmente d'autant. Cela a le même effet que :

```
$array[] = $var;
```

repeté pour chaque *var*.

Retourne le nouveau nombre d'éléments du tableau.

Exemple avec [array_push\(\)](#)

```
$stack = array (1, 2);
array_push($stack, "+", 3);
```

Cet exemple fait que \$stack a 4 éléments: 1, 2, "+", et 3.

Voir aussi: [array_pop\(\)](#), [array_shift\(\)](#), et [array_unshift\(\)](#). Note : *Cette fonction a été ajoutée dans PHP 4.0.*

10.2.13 array_reverse

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_reverse](#)(array *array*)

[array_reverse\(\)](#) prend le tableau *array* et retourne un nouveau tableau qui contient les mêmes éléments mais dans l'ordre inverse.

Exemple avec [array_reverse\(\)](#)

```
$input = array ("php", 4.0, array ("rouge", "vert"));
$result = array_reverse ($input);
```

Au final, \$result contient (array ("rouge", "vert"), 4.0, "php"). Note : *Cette fonction a été ajoutée dans PHP 4.0 Beta 3.*

10.2.14 array_rand

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [array_rand](#)(array *input*, int *num_req*)

[array_rand\(\)](#) est pratique lorsque vous voulez sélectionner une ou plusieurs valeurs au hasard dans un tableau. Le paramètre *input* est un tableau, et *num_req* spécifie le nombre de valeurs que vous voulez obtenir (par défaut, c'est 1).

Si vous ne demandez qu'une entrée, [array_rand\(\)](#) retourne l'index de la valeur. Sinon, elle retourne un tableau d'index. Cela vous permet de faire une sélection au hasard de clés, ou bien de valeur.

N'oubliez pas d'appeler [srand\(\)](#) pour initialiser le générateur de nombres aléatoires.

Exemple avec [array_rand\(\)](#)

```
srand ((double) microtime() * 10000000);
$input = array ("Neo", "Morpheus", "Trinitée", "Cypher", "Tank");
$rand_keys = array_rand ($input, 2);
print $input[$rand_keys[0]]."\n";
print $input[$rand_keys[1]]."\n";
```

[10.2.15 array_shift](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [array_shift](#) (array *array*)

[array_shift\(\)](#) extrait la première valeur d'un tableau et la retourne, en raccourcissant le tableau d'un élément, et en déplaçant tous les éléments vers le bas.

Exemple avec [array_shift\(\)](#)

```
$args = array ("-v", "-f");
$opt = array_shift ($args);
```

Cet exemple aura pour résultat que \$args ne contiendra plus que "-f", et \$opt contient "-v".

Voir aussi [array_unshift\(\)](#), [array_push\(\)](#), et [array_pop\(\)](#). Note : *Cette fonction a été ajoutée dans PHP 4.0.*

[10.2.16 array_slice](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_slice](#) (array *array*, int *offset*, int *length*)

[array_slice\(\)](#) retourne une série d'éléments du tableau *array* commençant à l'offset *offset* et représentant *length* éléments.

Si *offset* est positif, la série commencera à cet offset dans le tableau *array*. Si *offset* est négatif, cette série commencera à l'offset *offset* mais en commençant à la fin du tableau *array*.

Si *length* est donné et positif, alors la série aura autant d'éléments. Si *length* est donné et négatif, les éléments seront pris dans l'ordre inverse. Si *length* est omis, la séquence lira tous les éléments du tableau, depuis l'*offset* précisé jusqu'à la fin du tableau.

Exemple avec [array_slice\(\)](#)

```
$input = array ("a", "b", "c", "d", "e");
$output = array_slice ($input, 2); // retourne "c", "d", et "e"
$output = array_slice ($input, 2, -1); // retourne "c", "d"
$output = array_slice ($input, -2, 1); // retourne "d"
$output = array_slice ($input, 0, 3); // retourne "a", "b", et "c"
```

Voir aussi [array_splice\(\)](#). Note : *Cette fonction a été ajoutée dans PHP 4.0.*

[10.2.17 array_splice](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_splice](#) (array *input*, int *offset*, int *length* , array *replacement*)

[array_splice\(\)](#) supprime les éléments désignés par *offset* et *length* du tableau *input* et les remplace par les

éléments du tableau *remplacement*, si ce dernier est présent.

Si *offset* est positif, la série commencera à cet offset dans le tableau *input*. Si *offset* est négatif, cette série commencera à l'offset *offset* mais en commençant à la fin du tableau *input*.

Si *length* est donné et positif, alors la série aura autant d'éléments. Si *length* est donné et négatif, les éléments seront pris dans l'ordre inverse. Si *length* est omis, la séquence lira tous les éléments du tableau, depuis l'offset précisé jusqu'à la fin du tableau. Conseil : pour supprimer tous les éléments du tableau depuis *offset* jusqu'à la fin, même si un tableau de remplacement *remplacement* est spécifié, utilisez `count(count($input))` à la place de *length*.

Si *remplacement* est précisé, alors les éléments supprimés sont remplacés par les éléments de ce tableau. Si *offset* et *length* sont tels que la taille du tableau ne change pas, alors les éléments du tableau de remplacement *remplacement* sont insérés à partir de l'offset *offset*. Conseil : si le tableau de remplacement ne contient qu'un seul élément, il n'est pas obligatoire de forcer le type à array avec `array()`, à moins que cette variable ne soit elle même un tableau.

Les codes suivants sont équivalents :

```
array_push($input, $x, $y)    array_splice($input, count($input), 0, array($x, $y))
array_pop($input)           array_splice($input, -1)
array_shift($input)         array_splice($input, 0, 1)
array_unshift($input, $x, $y) array_splice($input, 0, 0, array($x, $y))
$a[$x] = $y                 array_splice($input, $x, 1, $y)
```

Retourne le tableau des éléments supprimés.

Exemples avec [array_splice\(\)](#)

```
$input = array("red", "green", "blue", "yellow");
array_splice($input, 2);      // $input est array("red", "green")
array_splice($input, 1, -1); // $input est array("red", "yellow")
array_splice($input, 1, count($input), "orange");
                             // $input est array("red", "orange")
array_splice($input, -1, 1, array("black", "maroon"));
                             // $input est array("red", "green",
                             //                    "blue", "black", "maroon")
```

Voir aussi [array_slice\(\)](#). Note : Cette fonction a été ajoutée dans PHP 4.0.

10.2.18 [array_unique](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_unique](#) (array *array*)

[array_unique\(\)](#) prend le tableau *array* et retourne un nouveau tableau, complètement dédoublonné. Notez que les clés sont préservées.

Exemple avec [array_unique\(\)](#)

```
$input = array ("a" => "vert", "rouge", "b" => "vert", "bleu", "rouge");
$result = array_unique ($input);
```

\$result contient `array ("a" => "vert", "rouge", "bleu");`

10.2.19 array_unshift

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [array_unshift](#) (array *array*, mixed *var*, ...)

[array_unshift\(\)](#) ajoute les éléments passé en argument au début du tableau *array*. Notez que les éléments sont ajoutés comme un tout, et qu'ils restent dans le même ordre.

Retourne le nouveau nombre d'éléments du tableau *array*.

Exemples avec [array_unshift\(\)](#)

```
$queue = array("p1", "p3");
array_unshift($queue, "p4", "p5", "p6");
```

Le résultat de cet exemple est que `$queue` aura 5 éléments, à savoir: "p4", "p5", "p6", "p1", et "p3".

Voir aussi [array_shift\(\)](#), [array_push\(\)](#), et [array_pop\(\)](#). Note : *Cette fonction a été ajoutée dans PHP 4.0.*

10.2.20 array_values

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_values](#) (array *input*)

[array_values\(\)](#) retourne les valeurs du tableau *input*.

Exemples avec [array_values\(\)](#)

```
$array = array("taille" => "XL", "couleur" => "or");
array_values($array); // // retourne array("XL", "or")
```

Note : *Cette fonction a été ajoutée dans PHP 4. Ci dessous, voici une implémentation pour ceux qui utilisent toujours PHP 3. implémentation de [array_values\(\)](#) pour les utilisateurs php 3 implémentation de [array_values\(\)](#) pour les utilisateurs php 3*

```
function array_values ($arr) {
    $t = array();
    while (list($k, $v) = each ($arr)) {
        $t[] = $v;
    }
    return $t;
}
```

10.2.21 [array_walk](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [array_walk](#)(array *arr*, string *func*, mixed *userdata*)

Exécute la fonction *func* avec chaque élément du tableau *arr*. Les éléments sont passés en tant que premier argument de la fonction *func*;

Si *func* a besoin de plus d'un argument, une alerte sera générée pour chaque appel de *func*. Ces alertes sont supprimées en ajoutant le suffixe " avant l'appel de [array_walk\(\)](#) ou simplement en utilisant [error_reporting\(\)](#).

Note : Si *func* doit travailler avec les véritables valeur du tableau, spécifiez que le premier paramètre de *func* doit être passé par référence. Alors, les éléments seront directement modifiés dans le tableau.

Note : Passez les clés et *userdata* à *func* a été ajouté dans PHP 4.0.

Dans PHP 4, [reset\(\)](#) doit être appelés si nécessaire, car [array_walk\(\)](#) ne réinitialise pas automatiquement le tableau.

Exemple avec [array_walk\(\)](#)

```
$fruits = array("d"=>"citron", "a"=>"orange", "b"=>"banane", "c"=>"pomme");
function test_alter( $item1 ) {
    $item1 = 'bidon';
}
function test_print( $item2 ) {
echo "$item2<br>\n";
}
array_walk( $fruits, 'test_print' );
array_walk( $fruits, 'test_alter' );
array_walk( $fruits, 'test_print' );
```

Voir aussi [each\(\)](#) et [list\(\)](#).

10.2.22 [arsort](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [arsort](#)(array *array*)

Cette fonction trie un tableau de telle manière que la corrélation entre les index et les valeurs soient conservées. L'usage principal est lors de tri de tableaux associatifs où l'ordre des éléments est important.

Exemple avec [arsort\(\)](#)

```
$fruits = array("d"=>"papaye", "a"=>"orange", "b"=>"banane", "c"=>"ananas");
arsort( $fruits);
for (reset( $fruits); $key = key( $fruits); next( $fruits)) {
echo "fruits[$key] = ".$fruits[$key]."\n";
}
```

Cet exemple va afficher: fruits[a] = orange fruits[d] = papaye fruits[b] = banane fruits[c] = ananas Les fruits ont été triés en ordre alphabétique inverse, et leur index respectifs ont été conservé.

Voir aussi: [asort\(\)](#), [rsort\(\)](#), [ksort\(\)](#), et [sort\(\)](#).

10.2.23 asort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [asort\(\)](#) (array *array*)

Cette fonction trie un tableau de telle manière que la corrélation entre les index et les valeurs soient conservées. L'usage principal est lors de tri de tableaux associatifs où l'ordre des éléments est important.

Exemple avec [asort\(\)](#)

```
$fruits = array("d"=>"papaye", "a"=>"orange", "b"=>"banane", "c"=>"ananas");
asort($fruits);
for(reset($fruits); $key = key($fruits); next($fruits)) {
echo "fruits[$key] = ".$fruits[$key]."\n";
}
```

Cet exemple va afficher: fruits[c] = ananas fruits[b] = banane fruits[d] = papaye fruits[a] = orange Les fruits ont été triés par ordre alphabétique, et leur index respectifs ont été conservé.

Voir aussi [arsort\(\)](#), [rsort\(\)](#), [ksort\(\)](#), et [sort\(\)](#).

10.2.24 compact

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [compact\(\)](#) (*string varname*) array *varnames*, ... |

[compact\(\)](#) accepte différents paramètres. Les paramètres peuvent être des variables contenant des chaînes, ou un tableau de chaîne, qui peut contenir d'autres tableau de noms, que [compact\(\)](#) traitera récursivement. Pour chacun des arguments, [compact\(\)](#) recherche une variable avec une variable de même nom dans la table courante des symboles, et l'ajoute dans le tableau, de manière à avoir la relation nom => 'valeur de variable'. En bref, c'est le contraire de la fonction [extract\(\)](#). Retourne le tableau ainsi créé.

Exemple avec [compact\(\)](#)

```
$ville = "San Francisco";
$etat = "CA";
$evenement = "SIGGRAPH";
$location_vars = array("ville", "etat");
$result = compact("evenement", $location_vars);
```

Après cette opération, \$result sera le tableau suivant : array(("evenement" => "SIGGRAPH", "ville" => "San Francisco", "etat" => "CA").

Voir aussi [extract\(\)](#). Note : Cette fonction a été ajoutée dans PHP 4.0.

10.2.25 count

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [count](#) (mixed *var*)

Retourne le nombre d'élément dans *var*, qui est généralement un tableau (et tout le reste n'aura qu'un élément).

Retourne 1 si la variable n'est pas un tableau.

Retourne 0 si la variable n'est pas créée.

[count\(\)](#) peut retourner 0 pour une variable qui n'a pas été affectée, ou pour un tableau vide. Utilisez plutôt [isset\(\)](#) pour tester si la variable existe.

Exemple avec [count\(\)](#)

```

$a[0] = 1;
$a[1] = 3;
$a[2] = 5;
$result = count ($a);
// $result == 3, pas 2, car il y a 3 éléments assignés
$a[2] = 1;
$a[4] = "";
$a[6] = 5;
$a[8] = 7;
$a[10] = 11;
$a[12] = "";
$result = count ($a);
// $result == 4, car il y a 4 éléments assignés

```

Voir aussi: [sizeof\(\)](#), [isset\(\)](#), et [is_array\(\)](#).

10.2.26 current

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [current](#) (array *array*)

Chaque tableau entretient un pointeur interne, qui est initialisé lors lorsque le premier élément est inséré dans le tableau.

La fonction [current\(\)](#) ne fait que retourner l'élément courant pointé par le pointeur interne. [current\(\)](#) ne déplace pas le pointeur. Si le pointeur est au delà du dernier élément de la liste, [current\(\)](#) retourne faux. Si le tableau des éléments vides ou des zéros (0 ou "", la chaîne vide) alors cette fonction retournera false pour ces éléments. Il est donc impossible de déterminer si vous êtes réellement à la fin de la liste en utilisant la fonction [current\(\)](#). Pour passer en revue proprement un tableau qui peut contenir des éléments vides ou des zéros, utilisez la fonction [each\(\)](#).

Voir aussi: [end\(\)](#), [next\(\)](#), [prev\(\)](#) et [reset\(\)](#).

10.2.27 [each](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [each](#)(array *array*)

Retourne la paire (clé/valeur) courante du tableau *array* et avance le pointeur de tableau. Cette paire est retournée dans un tableau de 4 éléments, avec les clés *0*, *1*, *key*, et *value*. Les éléments *0* et *key* contiennent le nom de la clé et, et *1* et *value* contiennent la valeur.

Si le pointeur interne de fichier est au delà de la fin du tableau, [each\(\)](#) retourne faux.

Exemples avec [each\(\)](#)

```
$foo = array ("bob", "fred", "jussi", "jouni", "egon", "marliese");
$bar = each ($foo);
```

\$bar contient maintenant les paires suivantes:

- 0 => 0
- 1 => 'bob'
- key => 0
- value => 'bob'

```
$foo = array ("Robert" => "Bob", "Seppo" => "Sepi");
$bar = each ($foo);
```

\$bar contient maintenant les paires suivantes:

- 0 => 'Robert'
- 1 => 'Bob'
- key => 'Robert'
- value => 'Bob'

[each\(\)](#) est utilisé conjointement avec [list\(\)](#) pour étudier tous les éléments d'un tableau; par exemple, \$HTTP_POST_VARS: affichage de \$http_post_vars avec [each\(\)](#) *affichage de \$http_post_vars avec [each\(\)](#)*

```
echo "Valeurs transmises par la méthode POST:<br>";
reset ($HTTP_POST_VARS);
while (list ($key, $val) = each ($HTTP_POST_VARS)) {
echo "$key => $val<br>";
}
```

Après chaque [each\(\)](#), le pointeur de tableau est déplacé au dernier éléments, ou sur le dernier élément, lorsqu'on arrive à la fin.

Voir aussi [key\(\)](#), [list\(\)](#), [current\(\)](#), [reset\(\)](#), [next\(\)](#), et [prev\(\)](#).

10.2.28 end

[\[Notes en ligne\]](#) [\[Exemples\]](#)

end (array *array*)

[end\(\)](#) déplace le pointeur interne du tableau *array* jusqu'au dernier élément.

Voir aussi: [current\(\)](#), [each\(\)](#), [end\(\)](#), [next\(\)](#), et [reset\(\)](#).

10.2.29 extract

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [extract](#)(array *var_array*, int *extract_type* , string *prefix*)

Cette fonction sert à exporter un tableau vers la table des symboles. Elle prend un tableau associatif *var_array* et crée les variables dont les noms sont les index de ce tableau, et leur affecte la valeur associée. Pour chaque paire clé/valeur, cette fonction crée une variable, avec les paramètres *extract_type* et *prefix*. [extract\(\)](#) vérifie l'existence de la variable avant de la créer. La manière de traiter les collisions est déterminée par *extract_type*. Ce paramètre peut prendre une des valeurs suivantes :

EXTR_OVERWRITE

- Lors d'une collision, réécrire la variable existante.
EXTR_SKIP
- Lors d'une collision, ne pas réécrire la variable existante
EXTR_PREFIX_SAME
- Lors d'une collision, ajouter le préfixe *prefix*, et créer une nouvelle variable.
EXTR_PREFIX_ALL
- Ajouter le préfixe *prefix*, et créer une nouvelle variable.

Si *extract_type* est omis, [extract\(\)](#) utilise EXTR_OVERWRITE par défaut.

Notez que *prefix* n'est nécessaire que pour les valeurs de *extract_type* suivantes : EXTR_PREFIX_SAME et EXTR_PREFIX_ALL.

[extract\(\)](#) vérifie que les clés constitue un nom de variable valide, et si c'est le cas, procède à son exportation.

Une utilisation possible de cette fonction est l'exportation vers la table des symboles de tableau de variables retourné par la fonction [wddx_deserialize\(\)](#).

Exemples avec [extract\(\)](#)

```
<php?
/* Supposons que $var_array est un tableau retourné par
wddx_deserialize */
$taille = "grand";
$var_array = array("couleur" => "bleu",
                  "taille"  => "moyen",
                  "forme"  => "sphere");
extract($var_array, EXTR_PREFIX_SAME, "wddx");
print "$couleur, $taille, $forme, $wddx_taille\n";
?>
```

L'exemple ci dessus va afficher

```
bleu, large, sphere, moyen
```

La variable \$taille n'a pas été réécrite, car on avait spécifié le paramètre EXTR_PREFIX_SAME, qui a permis la création \$wddx_size. Si EXTR_SKIP avait été utilisé, alors \$wddx_size n'aurait pas été créé. Avec EXTR_OVERWRITE, \$taille aurait pris la valeur "moyen", et avec EXTR_PREFIX_ALL, les variables créées seraient \$wddx_couleur, \$wddx_taille, et \$wddx_forme.

10.2.30 in_array

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool `in_array` (mixed *needle*, array *haystack*)

Recherche *needle* dans *haystack* et retourne vrai si il s'y trouve, ou faux sinon.

Exemple avec [in_array\(\)](#)

```
$os = array("Mac", "NT", "Irix", "Linux");
if (in_array("Irix", $os))
print "Irix trouve";
```

Note : Cette fonction a été ajoutée dans PHP 4.0.

10.2.31 key

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed `key` (array *array*)

`key()` retourne l'index de la clé courante dans un tableau.

Voir aussi: [current\(\)](#), et [next\(\)](#)

10.2.32 krsort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int `krsort` (array *array*)

`krsort()` trie un tableau en ordre inverse et suivant les clés, en maintenant la correspondance entre les clés et les valeurs. Cette fonction est pratique pour les tableaux associatifs. *Exemple avec [krsort\(\)](#)*

```
$fruits = array("d"=>"papaye", "a"=>"orange", "b"=>"banane", "c"=>"ananas");
ksort($fruits);
for(reset($fruits); $key = key($fruits); next($fruits)) {
echo "fruits[$key] = ".$fruits[$key]."\n";
```

```
}
```

Cet exemple va afficher : fruits[d] = citron fruits[c] = ananas fruits[b] = banane fruits[a] = orange

Voir aussi [asort\(\)](#), [arsort\(\)](#), [ksort\(\)](#) [sort\(\)](#), et [rsort\(\)](#).

10.2.33 ksort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ksort\(\)](#)(array *array*)

[ksort\(\)](#) trie un tableau suivant les clés, en maintenant la correspondance entre les clés et les valeurs. Cette fonction est pratique pour les tableaux associatifs. *Exemple avec [ksort\(\)](#)*

```
$fruits = array("d"=>"papaye", "a"=>"orange", "b"=>"banane", "c"=>"ananas");
ksort ($fruits);
reset ($fruits);
while (list ($key, $val) = each ($fruits)) {
echo "$key -> $val\n";
}
```

Cet exemple va afficher : fruits[a] = orange fruits[b] = banane fruits[c] = ananas fruits[d] = citron

Vous pouvez modifier le comportement du tri avec les options *sort_flags*. Pour plus de détails, voyez [sort\(\)](#).

Voir aussi [asort\(\)](#), [arsort\(\)](#), [sort\(\)](#), et [rsort\(\)](#).

Note : *Le second paramètre a été ajouté dans PHP 4.*

10.2.34 list

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [list\(\)](#)

Tout comme [array\(\)](#), ce n'est pas une véritable fonction, mais une construction syntaxique, qui permet d'assigner une série de variable en une seule ligne. *Exemple avec [list\(\)](#)*

```
<table>
<tr>
<th>Employee name</th>
<th>Salary</th>
</tr>
<?php
$result = mysql ($conn, "SELECT id, name, salary FROM employees");
while (list ($id, $name, $salary) = mysql_fetch_row ($result)) {
print (" <tr>\n".
" <td><a href=\"info.php3?id=$id\">$name</a></td>\n".
" <td>$salary</td>\n".
" </tr>\n");
}
?>
```

</table>

Voir aussi: [each\(\)](#), [array\(\)](#).

10.2.35 natsort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [natsort\(\)](#)(array *array*)

Cette fonction implémente un algorithme de tri qui traite les chaînes alphanumériques comme un être humain : c'est ce qui est appelé l'"ordre naturel". Un exemple de la différence de traitement entre un tel algorithme et un algorithme de tri de chaîne (comme lorsqu'on utilise [sort\(\)](#)) est illustré ci dessous :

Exemple avec [natsort\(\)](#)

```
$array1 = $array2 = array ( "img12.png", "img10.png", "img2.png", "img1.png" );
sort($array1);
echo "Tri Standard\n";
print_r($array1);
natsort($array2);
echo "\nTri par Ordre Naturel\n";
print_r($array2);
```

L'exemple ci dessous génère l'affichage suivant :

```
Tri Standard
Array
(
    [0] => img1.png
    [1] => img10.png
    [2] => img12.png
    [3] => img2.png
)
Tri par Ordre Naturel
Array
(
    [3] => img1.png
    [2] => img2.png
    [1] => img10.png
    [0] => img12.png
)
```

Pour plus de détails, rendez vous sur le site de : Martin Pool's [Natural Order String Comparison](#).

Voir aussi [natcasesort\(\)](#), [@xref{function.strnatcmp}](#) et [@xref{function.strnatcasecmp}](#).

10.2.36 natcasesort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [natcasesort\(\)](#)(array *array*)

Cette fonction implémente un algorithme de tri qui traite les chaînes alphanumériques comme un être humain : c'est ce qui est appelé l'"ordre naturel".

[natcasesort\(\)](#) est la version insensible à la casse de [natsort\(\)](#). Voir aussi [natsort\(\)](#) pour un exemple illustré.

Pour plus de détails, rendez vous sur le site de : Martin Pool's [Natural Order String Comparison](#).

Voir aussi [sort\(\)](#), [natsort\(\)](#), [@xref{function.strnatcmp , , strnatcmp\(\)}](#) et [@xref{function.strnatcasecmp , , strnatcasecmp\(\)}](#).

10.2.37 next

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [next](#) (array *array*)

[next\(\)](#) retourne l'élément suivant du tableau, ou false si il n'y a plus d'éléments. Le pointeur de interne de tableau est avancé d'un élément.

[next\(\)](#) se comporte comme [current\(\)](#), mais avec une différence : il avance le pointeur interne de tableau d'un élément avant de retourner la valeur qu'il pointe. Lorsque le pointeur dépasse le dernier élément, [next\(\)](#) retourne false. Si le tableau contient des éléments vides ou des zéros, cette fonction retournera false pour ces éléments. Pour passer proprement en revue un tableau, il faut utiliser [each\(\)](#).

Voir aussi: [current\(\)](#), [end\(\)](#) [prev\(\)](#) et [reset\(\)](#)

10.2.38 pos

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [pos](#) (array *array*)

C'est une fonction alias de [current\(\)](#).

Voir aussi: [end\(\)](#), [next\(\)](#), [prev\(\)](#) et [reset\(\)](#).

10.2.39 prev

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [prev](#) (array *array*)

Repositionne le pointeur interne de tableau à la dernière place qu'il occupait, ou bien retourne faux si il ne reste plus d'éléments. Si le tableau contient des éléments vides, cette fonction retournera faux pour ces éléments aussi. Pour passer en revue tous les éléments, utilisez plutôt [each\(\)](#).

[prev\(\)](#) se comporte exactement comme [next\(\)](#), mais il fait reculer le pointeur plutôt que de l'avancer.

Voir aussi: [current\(\)](#), [end\(\)](#) [next\(\)](#) et [reset\(\)](#)

10.2.40 range

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [range](#) (int *low*, int *high*)

[range\(\)](#) retourne un tableau contenant tous les entiers depuis *low* jusqu'à *high*, inclus.

Voir [shuffle\(\)](#) pour un exemple d'utilisation.

[10.2.41 reset](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [reset](#)(array *array*)

[reset\(\)](#) replace le pointeur de tableau *array* au premier élément.

[reset\(\)](#) retourne la valeur du premier élément.

Voir aussi: [current\(\)](#), [each\(\)](#), [next\(\)](#), [prev\(\)](#), et [reset\(\)](#).

[10.2.42 rsort](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [rsort](#)(array *array*)

Cette fonction effectue un trie en ordre décroissant (du plus grand au plus petit). *Exemple avec [rsort\(\)](#)*

```
$fruits = array("papaye", "orange", "banane", "ananas");
rsort($fruits);
for (reset($fruits); list($key,$value) = each($fruits); ) {
echo "fruits[$key] = ", $value, "\n";
}
```

Cet exemple va afficher: fruits[0] = papaye fruits[1] = orange fruits[2] = banane fruits[3] = ananas Les fruits ont été classés dans l'ordre alphabétique inverse.

Voir aussi: [arsort\(\)](#), [asort\(\)](#), [ksort\(\)](#), [sort\(\)](#), et [usort\(\)](#).

[10.2.43 shuffle](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [shuffle](#)(array *array*)

Cette fonction mélange les éléments d'un tableau. *Exemple avec [shuffle\(\)](#)*

```
$numbers = range (1,20);
srand (time());
shuffle ($numbers);
while (list(, $number) = each ($numbers)) {
echo "$number ";
}
```

Voir aussi [arsort\(\)](#), [asort\(\)](#), [ksort\(\)](#), [rsort\(\)](#), [sort\(\)](#) and [usort\(\)](#).

[10.2.44 sizeof](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sizeof](#)(array *array*)

Retourne le nombre d'élément d'un tableau.

Voir aussi: [count\(\)](#)

[10.2.45 sort](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [sort](#)(array *array*)

Cette fonction trie le tableau *array*. Les éléments seront triés du plus petit au plus grand. *Exemple avec [sort\(\)](#)*

```
$fruits = array("papaye", "orange", "banane", "ananas");
sort($fruits);
for(reset($fruits); $key = key($fruits); next($fruits)) {
echo "fruits[$key] = ".$fruits[$key]."\n";
}
```

Cet exemple va afficher: fruits[0] = ananas fruits[1] = banane fruits[2] = orange fruits[3] = papaye Les fruits ont été classé dans l'ordre alphabétique.

Voir aussi: [arsort\(\)](#), [asort\(\)](#), [ksort\(\)](#), [rsort\(\)](#), et [usort\(\)](#).

[10.2.46 uasort](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [uasort](#)(array *array*, fonction *cmp_function*)

Cette fonction trie un tableau en conservant la correspondance entre les index et leurs valeurs. Cette fonction sert essentiellement lors de tri de tableaux associatifs où l'ordre des éléments est significatif. La fonction de comparaison utilisée est définie par l'utilisateur.

[10.2.47 uksort](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [uksort](#)(array *array*, fonction *cmp_function*)

[uksort\(\)](#) trie les clés du tableau en utilisant une fonction définie par l'utilisateur. Si un tableau qui doit être trié avec un critère complexe, il est préférable d'utiliser cette fonction. *Exemple avec [uksort\(\)](#)*

```
function mycompare($a, $b) {
if ($a == $b) return 0;
return ($a > $b) ? -1 : 1;
}
$a = array(4 => "quatre", 3 => "trois", 20 => "vingt", 10 => "dix");
uksort($a, mycompare);
while(list($key, $value) = each($a)) {
echo "$key: $value\n";
}
```

}

Cet exemple affichera: 20: vingt 10: dix 4: quatre 3: trois
 Voir aussi: [arsort\(\)](#), [asort\(\)](#), [uasort\(\)](#), [ksort\(\)](#), [rsort\(\)](#), et [sort\(\)](#).

10.2.48 usort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [usort](#) (array *array*, fonction *cmp_function*)

Cette fonction va trier un tableau avec ses valeurs, en utilisant une fonction définie par l'utilisateur. Si un tableau doit être trié avec un critère complexe, il est préférable d'utiliser cette méthode. La fonction de comparaison doit retourner un entier, qui sera inférieur, égal ou supérieur à zéro suivant que le premier argument est considéré comme plus petit, égal ou plus grand que le second argument. Si les deux arguments sont égaux, leur ordre est indéfini. *Exemple avec [usort\(\)](#)*

```
function cmp($a,$b) {
if ($a == $b) return 0;
return ($a > $b) ? -1 : 1;
}
$a = array(3,2,5,6,1);
usort($a, cmp);
while(list($key,$value) = each($a)) {
echo "$key: $value\n";
}
```

Cet exemple va afficher : 0 : 6 1 : 5 2 : 3 3 : 2 4 : 1 Note : *Evidemment dans ce cas trivial, [rsort\(\)](#) serait plus approprié.*

Les bibliothèques de tri rapides sur lesquelles reposent PHP peuvent le conduire à un plantage, si la fonction de comparaison ne retourne pas une valeur cohérente.

Voir aussi: [arsort\(\)](#), [asort\(\)](#), [ksort\(\)](#), [rsort\(\)](#) and [sort\(\)](#).

10.3 Fonctions Aspell

[\[Notes en ligne\]](#)

Les fonctions Aspell vous permettent de vérifier l'orthographe d'un mot, et d'offrir des suggestions de corrections. Plusieurs langues sont disponibles, comme le franç, l'allemand, le suédois et le danois. Note : *aspell fonctionne avec de très vieilles versions (jusqu'à la version .27.* ou presque) de la librairie aspell. Ce module, et ces versions d'aspell ne sont plus supportées. Si vous voulez utiliser les possibilités de vérifications d'orthographe en PHP, utilisez plutôt [10.56 Pspell Functions](#). Ce module utilise la librairie pspell qui fonctionne avec les nouvelles versions de aspell.*

Vous avez besoin de la librairie Aspell, disponible à : <http://aspell.sourceforge.net/>

10.3.1 aspell_new

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [aspell_new](#) (string *master*, string *personal*)

[aspell_new\(\)](#) ouvre un nouveau dictionnaire, et retourne un identifiant de dictionnaire pour utilisation ultérieure dans les fonctions aspell.

aspell_new

```
$aspell_link=aspell_new("english");
```

10.3.2 aspell_check

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [aspell_check](#)(int *dictionary_link*, string *word*)

[aspell_check\(\)](#) vérifie l'orthographe d'un mot et retourne TRUE si l'orthographe est correcte, et FALSE sinon.

aspell_check

```
$aspell_link=aspell_new("english");
if (aspell_check($aspell_link,"testt")) {
echo "L'orthographe est correcte.";
} else {
echo "Désolé, l'orthographe est incorrecte.";
}
```

10.3.3 aspell_check_raw

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [aspell_check_raw](#)(int *dictionary_link*, string *word*)

[aspell_check_raw\(\)](#) vérifie l'orthographe d'un mot sans en changer la casse, et sans essayer de supprimer les espaces aux extrémités. Elle retourne true si l'orthographe est bonne, et false sinon.

aspell_check_raw

```
$aspell_link=aspell_new("english");
if (aspell_check_raw($aspell_link,"testt")) {
echo "This is a valid spelling";
} else {
echo "Sorry, wrong spelling";
}
```

10.3.4 `aspell_suggest`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array `aspell_suggest`(int *dictionary_link*, string *word*)

`aspell_suggest()` retourne un tableau contenant les orthographes possibles d'un mot mal formé.
aspell_suggest

```
$aspell_link=aspell_new("english");
if (!aspell_check($aspell_link,"testt")) {
    $suggestions=aspell_suggest($aspell_link,"testt");
    for($i=0; $i < count($suggestions); $i++) {
        echo "Orthographe envisageable : " . $suggestions[$i] . "<br>";
    }
}
```

10.4 Fonctions mathématiques sur des nombres de taille arbitraire

[\[Notes en ligne\]](#)

Ces fonctions ne sont disponibles que si l'option de configuration `—enable-bcmath` a été activée lors de la compilation.

10.4.1 `bcadd`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string `bcadd`(string *left operand*, string *right operand*, int *scale*)

Additionne *left operand* avec l'opérande *right operand* et renvoie la somme sous forme de chaîne de caractères. Le paramètre optionnel *scale* est utilisé pour définir le nombre de chiffres après la virgule dans le résultat.

Voir aussi `bcsub()`.

10.4.2 `bccomp`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int `bccomp`(string *left operand*, string *right operand*, int *scale*)

Compare l'opérande *left operand* avec l'opérande *right operand* et renvoie le résultat sous forme de valeur numérique (integer). Le paramètre optionnel *scale* est utilisé pour définir le nombre de chiffres après la virgule utilisés lors de la comparaison. Le résultat est 0 si les deux opérandes sont égales. Si l'opérande *left operand* est plus grande que l'opérande *right operand*, le résultat est 1. Si l'opérande *left operand* est plus petite que l'opérande *right operand*, le résultat est -1.

[10.4.3 bcdiv](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bcdiv](#) (string *left operand*, string *right operand*, int *scale*)

Divise l'opérande *left operand* par l'opérande *right operand* et renvoie le résultat. Le paramètre optionnel *scale* définit le nombre de chiffres après la virgule dans le résultat.

Voir aussi [bcmul\(\)](#).

[10.4.4 bcmath](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bcmmod](#) (string *left operand*, string *modulus*)

Retourne le reste de la division entre *left operand* en utilisant *modulus*.

Voir aussi [bcdiv\(\)](#).

[10.4.5 bcmul](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bcmul](#) (string *left operand*, string *right operand*, int *scale*)

Multiplie l'opérande *left operand* par l'opérande *right operand* et renvoie le résultat. Le paramètre optionnel *scale* définit le nombre de chiffres après la virgule dans le résultat.

Voir aussi [bcdiv\(\)](#).

[10.4.6 bcpow](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bcpow](#) (string *x*, string *y*, int *scale*)

Elève *x* à la puissance *y*. Le paramètre optionnel *scale* définit le nombre de chiffres après la virgule dans le résultat.

Voir aussi [bcsqrt\(\)](#).

[10.4.7 bcscale](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bcscale](#) (int *scale*)

Cette fonction définit la précision par défaut pour toutes les fonctions mathématiques sur des nombres de taille arbitraire qui suivent et qui omettent le paramètre *scale*.

[10.4.8 bcsqrt](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bcsqrt](#) (string *operand*, int *scale*)

Renvoie la racine carrée de l'opérande *operand*. Le paramètre optionnel *scale* définit le nombre de chiffres après la virgule dans le résultat.

Voir aussi [bcpow\(\)](#).

10.4.9 bcsb

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bcsb](#) (string *left operand*, string *right operand*, int *scale*)

Soustrait l'opérande *right operand* de l'opérande *left operand* et renvoie le résultat sous forme de chaîne de caractères. Le paramètre optionnel *scale* définit le nombre de chiffres après la virgule dans le résultat.

Voir aussi [bcadd\(\)](#).

10.5 Fonctions de calendrier

[\[Notes en ligne\]](#)

Les fonctions de calendrier ne sont disponibles que si l'extension calendrier (calendar) a été compilée. Elle est située dans dl/calendar. Lisez le fichier dl/README pour plus de détails.

L'extension de calendrier propose une série de fonctions qui simplifie les conversions entre les différents formats de calendrier. La référence est le nombre de jour du calendrier Julien. C'est le nombre de jours depuis une date qui commence bien au delà des dates les plus reculées dont on a besoin (située en 4000 avant JC).

Pour convertir une date d'un calendrier à un autre, il faut d'abord la convertir dans ce calendrier, puis convertir le résultat dans le calendrier désiré. Attention, le nombre de jour du calendrier Julien est un système très différent du calendrier Julien !. Pour plus d'informations (en anglais), reportez vous à <http://genealogy.org/~scottlee/cal-overview.html>. Les traductions issues de ces pages seront mises entre guillemets.

10.5.1 JDToGregorian

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [jdtogregorian](#) (int *julianday*)

Converti le nombre de jours du calendrier Julien en une chaîne contenant une date du calendrier grégorien, au format "mois/jour/année".

10.5.2 GregorianToJD

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gregoriantojd](#) (int *month*, int *day*, int *year*)

Intervalle de validité pour le calendrier grégorien : 4714 avant JC à 9999 après JC.A.D.

Bien qu'il soit possible de manipuler des dates jusqu'en 4714 avant JC, une telle utilisation n'est pas significative. En effet, ce calendrier fut créé le 18 octobre 1582 après J.C. (ou 5 octobre 1582 en calendrier grec). Certains pays ne l'acceptèrent que bien plus tard. Par exemple, les britanniques n'y passèrent en 1752, les Russes en 1918 et les Grecs en 1923. La plus part des pays européens utilisaient le calendrier Julien avant

le Grégorien. *Fonctions calendrier*

```
<?php
$jd = GregorianToJD(10,11,1970);
echo("$jd\n");
$gregorian = JDToGregorian($jd);
echo("$gregorian\n");
?>
```

10.5.3 JDToJulian

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [jdtोजulian](#) (int *julianday*)

Converti le nombre de jours du calendrier Julien en une chaîne contenant la date du calendrier Julien, au format "mois/jour/année".

10.5.4 JulianToJD

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [juliantojd](#) (int *month*, int *day*, int *year*)

Intervalle de validité du calendrier Julien : 4713 avant JC à 9999 après J.C..

Bien qu'il soit possible de manipuler des dates jusqu'en 4713 avant JC, une telle utilisation n'est pas significative. En effet, ce calendrier fut créé en 46 avant JC, et ses détails ne furent finalisés qu'au plus tôt en 8 après JC, et probablement pas avant le 4ème siècle après JC. De plus, le début de l'année variait suivant les peuples, certains n'acceptant pas janvier comme premier mois de l'année.

10.5.5 JDToJewish

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [jdtोजewish](#) (int *julianday*)

Converti le nombre de jours du calendrier julien en date du calendrier juif.

10.5.6 JewishToJD

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [jewishtojd](#) (int *month*, int *day*, int *year*)

Bien qu'il soit possible de manipuler des dates à partir de l'an 1 (3761 avant JC), une telle utilisation a peu de sens.

Le calendrier juif a été utilisé depuis plusieurs dizaines de siècles, mais dans les premiers temps, il n'y avait pas de formule pour déterminer le début du mois. Un nouveau mois commençait quand une nouvelle lune était observée.

10.5.7 JDToFrench

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [jdtofrench](#) (int *month*, int *day*, int *year*)

Converti le nombre de jours du calendrier julien en date du calendrier français républicain.

10.5.8 FrenchToJD

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [frenchtojd](#) (int *month*, int *day*, int *year*)

Converti une date du calendrier français républicain en nombre de jour du calendrier julien. Ces fonctions convertissent les dates comprises entre l'an 1 et l'an 14 (22 September 1792 à 22 September 1806 en calendrier grégorien). Cela couvre plus que la durée d'existence de ce calendrier.

10.5.9 JDMonthName

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [jdmmonthname](#) (int *julianday*, int *mode*)

Retourne une chaîne contenant le nom du mois. *mode* indique de quel calendrier dépend ce mois, et quel type de nom doit être retourné.

Mode	Signification
0	Grégorien – abrégé
1	Grégorien
2	Julien – abrégé
3	Julien
4	Juif
5	Républicain français

10.5.10 JDDayOfWeek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [jddayofweek](#) (int *julianday*, int *mode*)

Retourne le numéro du jour de la semaine. Peut retourner une chaîne ou un entier, en fonction du mode.

Mode	Signification
0	Retourne le numéro du jour comme un entier (0=dimanche, 1=lundi, etc.) @tab
1	Retourne une chaîne contenant le nom du jour (anglais grégorien) @tab
2	Retourne une chaîne contenant le nom abrégé du jour de la semaine (anglais grégorien). @tab

10.5.11 easter_date

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int easter_date(int *year*)

Retourne un timestamp UNIX pour P'ques, à minuit, pour une année donnée. Si l'année n'est pas précisée, c'est l'année en cours qui est utilisée.

ATTENTION: cette fonction génère une alerte (Warning) si la date tombe hors de la zone de validité des timestamps UNIX (i.e. avant 1970 ou après 2037). *Exemples avec easter_date()*

```
echo date( "M-d-Y", easter_date(1999) );          /* "04 avril 1999" */
echo date( "M-d-Y", easter_date(2000) );          /* "23 avril 2000" */
echo date( "M-d-Y", easter_date(2001) );          /* "15 avril 2001" */
```

La date de P'ques a été fixée par le concile de Nicée, en 325 de notre ère, comme étant le dimanche après la première lune pleine qui suit l'équinoxe de printemps. L'équinoxe de printemps est considéré comme étant toujours le 21 mars, ce qui réduit le problème au calcul de la date de la lune pleine qui suit, et le dimanche suivant. L'algorithme fut introduit vers 532, par Dionysius Exiguus. Avec le calendrier Julien, (pour les années avant 1753), un cycle de 19 ans suffit pour connaître les date des phases de la lune. Avec le calendrier grégorien, (à partir des années 1753, conçu par Clavius et Lilius, puis introduit par le pape Gregoire XIII en Octobre 1582, et en Grande Bretagne et ses colonies en septembre 1752), deux facteurs de corrections ont été ajoutés pour rendre le cycle plus précis.

(Ce code est basé sur le programme en C de Simon Kershaw, <webmaster@ely.anglican.org>)

Voir easter_days() pour les calculs de date de P'ques avant 1970 et après 2037.

10.5.12 easter_days

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int easter_days(int *year*)

Retourne le nombre de jour entre le 21 Mars et P'ques, pour une année donnée. Si l'année n'est pas précisée, l'année en cours est utilisée par défaut.

Cette fonction peut être utilisée à la place de easter_date() pour calculer la date de P'ques, pour les années qui tombent hors de l'intervalle de validité des timestamps UNIX (i.e. avant 1970 ou après 2037). *Exemple easter_date()*

```
echo easter_days(1999);          /* 14, i.e. 4 Avril */
echo easter_days(1492);          /* 32, i.e. 22 Avril */
echo easter_days(1913);          /* 2, i.e. 23 Mars */
```

La date de P'ques a été fixée par le concile de Nicée, en 325 de notre ère, comme étant le dimanche après la première lune pleine qui suit l'équinoxe de printemps. L'équinoxe de printemps est considéré comme étant toujours le 21 mars, ce qui réduit le problème au calcul de la date de la lune pleine qui suit, et le dimanche suivant. L'algorithme fut introduit vers 532, par Dionysius Exiguus. Avec le calendrier Julien, (pour les années avant 1753), un cycle de 19 ans suffit pour connaître les date des phases de la lune. Avec le calendrier

grégorien, (à partir des années 1753, conçu par Clavius et Lilius, puis introduit par le pape Gregoire XIII en Octobre 1582, et en Grande Bretagne et ses colonies en septembre 1752), deux facteurs de corrections ont été ajoutés pour rendre le cycle plus précis.

(Ce code est basé sur le programme en C de Simon Kershaw, <webmaster@ely.anglican.org>)

Voir aussi [easter_date\(\)](#).

10.6 CCVS API Functions

[\[Notes en ligne\]](#)

These functions interface the CCVS API, allowing you to directly work with CCVS from your PHP scripts. CCVS is [RedHat's](#) solution to the "middle-man" in credit card processing. It lets you directly address the credit card clearing houses via your *nix box and a modem. Using the CCVS module for PHP, you can process credit cards directly through CCVS via your PHP Scripts. The following references will outline the process.

To enable CCVS Support in PHP, first verify your CCVS installation directory. You will then need to configure PHP with the `--with-ccvs` option. If you use this option without specifying the path to your CCVS installation, PHP Will attempt to look in the default CCVS Install location (`/usr/local/ccvs`). If CCVS is in a non-standard location, run configure with: `--with-ccvs=$ccvs_path`, where `$ccvs_path` is the path to your CCVS installation. Please note that CCVS support requires that `$ccvs_path/lib` and `$ccvs_path/include` exist, and include `cv_api.h` under the include directory and `libccvs.a` under the lib directory.

Additionally, a `ccvsd` process will need to be running for the configurations you intend to use in your PHP scripts. You will also need to make sure the PHP Processes are running under the same user as your CCVS was installed as (e.g. if you installed CCVS as user 'ccvs', your PHP processes must run as 'ccvs' as well.)

Additional information about CCVS can be found at <http://www.redhat.com/products/ccvs>.

This documentation section is being worked on. Until then, RedHat maintains slightly outdated but still useful documentation at <http://www.redhat.com/products/ccvs/support/CCVS3.3docs/ProgPHP.html>.

@xref{function. , , () }

10.7 Fonctions Classe/Objet

[\[Notes en ligne\]](#)

10.7.1 get class methods

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [get_class_methods](#)(string *class_name*)

[get_class_methods\(\)](#) retourne un tableau contenant les noms des méthodes de la classe *class_name*.

10.7.2 get class vars

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [get_class_vars](#)(string *class_name*)

[get_class_vars\(\)](#) retourne un tableau contenant les valeurs par défaut des attributs de la classe *class_name*.

10.7.3 get_object_vars

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [get_object_vars](#)(object *obj*)

[get_object_vars\(\)](#) retourne un tableau contenant les valeurs des attributs de la classe *class_name*.

10.7.4 method_exists

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [method_exists](#)(object *object*, string *method_name*)

[method_exists\(\)](#) retourne TRUE si la méthode *method_name* a été définie pour la classe *object*, et sinon, retourne FALSE.

10.8 Support COM pour Windows

[\[Notes en ligne\]](#)

Ces fonctions ne sont disponibles que sous les versions Windows de PHP. Elles ont été ajoutées dans PHP4.

10.8.1 com_load

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [com_load](#)(string *module name*, string *server name*)

10.8.2 com_invoke

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [com_invoke](#)(resource *object*, string *function_name*, mixed *paramètres de fonction, ...*)

10.8.3 com_propget

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [com_propget](#)(resource *object*, string *property*)

10.8.4 com_get

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [com_get](#) (resource *object*, string *property*)

10.8.5 com_propput

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [com_propput](#) (resource *object*, string *property*, mixed *value*)

10.8.6 com_propset

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [com_propset](#) (resource *object*, string *property*, mixed *value*)

Cette fonction est un alias de [com_propput\(\)](#).

10.8.7 com_set

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [com_set](#) (resource *object*, string *property*, mixed *value*)

Cette fonction est un alias de [com_set\(\)](#).

10.9 Fonctions ClibPDF

[\[Notes en ligne\]](#)

ClibPDF vous permet de créer des documents PDF avec PHP. Cette librairie est disponible à [FastIO](#) mais n'est pas gratuite. Vous devez lire la licence avant de l'utiliser. Si vous ne pouvez pas accepter la licence, essayez plutôt [pdflib](#) de Thomas Merz, qui est aussi très puissante. Les fonctionnalités de ClibPDF et ses API sont très similaires à celles de Thomas Merz's [pdflib](#) mais, selon [FastIO](#), ClibPDF est plus rapide, et crée des documents plus compacts. Cela peut avoir changé depuis la version 2.0 de [pdflib](#). Un test de vitesse (avec [pdfclock.c](#) issue des exemples de [pdflib](#) 2.0 transformé en script PHP) ne montre aucune différence de vitesse. La taille des fichiers est similaire si la compression n'est pas utilisée. Il vaut mieux alors essayer les deux, et choisir celui qui vous convient le mieux.

Cette documentation devrait être lue avec le manuel ClibPDF sous la main, car il est beaucoup plus détaillé. Beaucoup de fonctions sont natives de ClibPDF et se retrouvent dans le module PHP `module`, et tout comme [pdflib](#), elles ont le même nom. Toutes les fonctions, hormis [cpdf_open\(\)](#) utilisent un pointeur sur un document comme premier paramètre. Actuellement, ce pointeur n'est pas utilisé en interne, car ClibPDF ne supporte pas la création de plusieurs documents PDF simultanément. En fait, il ne vaut mieux pas l'envisager, car les résultats sont aléatoires. Je ne veux même pas imaginer les problèmes qui pourrait se poser avec les environnements multi-tâches. Selon l'auteur de ClibPDF, cette situation va changer dans les prochaines versions (lorsque cette documentation a été traduite, c'était la version 1.10). Si vous avez besoin de cette

fonctionnalité, utilisez `pdflib`.

Note : La fonction `cpdf_set_font()` a changé depuis le PHP 3.0 pour supporter les polices asiatiques. Le paramètre d'encodage n'est plus un entier, mais une chaîne.

Un des gros avantages de `ClibPDF` sur `pdflib` est la possibilité de créer complètement un document sans passer par des fichiers temporaires. Il est aussi possible d'utiliser des coordonnées avec une unité de longueur prédéfinie. C'est une fonctionnalité bien pratique mais qui peut être simulée avec `pdf_translate()`.

La plus part des fonctions sont très simples d'emploi. Le plus difficile est probablement de créer un document PDF simple. L'exemple suivant devrait vous aider à démarrer. La page contient du texte qui utilise la police "Times–Roman" en taille 30, outlined. Le texte est souligné.

Exemple simple ClibPDF

```
<?php
$cpdf = cpdf_open(0);
cpdf_page_init($cpdf, 1, 0, 595, 842);
cpdf_add_outline($cpdf, 0, 0, 0, 1, "Page 1");
cpdf_set_font($cpdf, "Times–Roman", 30, "WinAnsiEncoding");
cpdf_set_text_rendering($cpdf, 1);
cpdf_text($cpdf, "Times Roman outlined", 50, 750);
cpdf_moveto($cpdf, 50, 740);
cpdf_lineto($cpdf, 330, 740);
cpdf_stroke($cpdf);
cpdf_finalize($cpdf);
Header("Content-type: application/pdf");
cpdf_output_buffer($cpdf);
cpdf_close($cpdf);
?>
```

La distribution `pdflib` contient un exemple plus complet, qui crée des séries de pages avec une horloge. Voici cet exemple converti en script PHP qui utilise l'extension `ClibPDF` :

Exemple pdfclock de la distribution pdflib 2.0

```
<?php
$radius = 200;
$margin = 20;
$pagecount = 40;
$pdf = cpdf_open(0);
cpdf_set_creator($pdf, "pdf_clock.php3");
cpdf_set_title($pdf, "Analog Clock");
while($pagecount-- > 0) {
cpdf_page_init($pdf, $pagecount+1, 0, 2 * ($radius + $margin), 2 * ($radius + $margin), 1.0);
cpdf_set_page_animation($pdf, 4, 0.5, 0, 0, 0); /* wipe */
cpdf_translate($pdf, $radius + $margin, $radius + $margin);
cpdf_save($pdf);
cpdf_setrgbcolor($pdf, 0.0, 0.0, 1.0);
/* minute strokes */
cpdf_setlinewidth($pdf, 2.0);
for ($alpha = 0; $alpha < 360; $alpha += 6)
{
cpdf_rotate($pdf, 6.0);
cpdf_moveto($pdf, $radius, 0.0);
cpdf_lineto($pdf, $radius-$margin/3, 0.0);
cpdf_stroke($pdf);
}
cpdf_restore($pdf);
```

```

cpdf_save($pdf);
/* 5 minute strokes */
cpdf_setlinewidth($pdf, 3.0);
for ($alpha = 0; $alpha < 360; $alpha += 30)
{
cpdf_rotate($pdf, 30.0);
cpdf_moveto($pdf, $radius, 0.0);
cpdf_lineto($pdf, $radius-$margin, 0.0);
cpdf_stroke($pdf);
}
$time = getdate();
/* draw hour hand */
cpdf_save($pdf);
cpdf_rotate($pdf, -((($time['minutes']/60.0) + $time['hours'] - 3.0) * 30.0));
cpdf_moveto($pdf, -$radius/10, -$radius/20);
cpdf_lineto($pdf, $radius/2, 0.0);
cpdf_lineto($pdf, -$radius/10, $radius/20);
cpdf_closepath($pdf);
cpdf_fill($pdf);
cpdf_restore($pdf);
/* draw minute hand */
cpdf_save($pdf);
cpdf_rotate($pdf, -((($time['seconds']/60.0) + $time['minutes'] - 15.0) * 6.0));
cpdf_moveto($pdf, -$radius/10, -$radius/20);
cpdf_lineto($pdf, $radius * 0.8, 0.0);
cpdf_lineto($pdf, -$radius/10, $radius/20);
cpdf_closepath($pdf);
cpdf_fill($pdf);
cpdf_restore($pdf);
/* draw second hand */
cpdf_setrgbcolor($pdf, 1.0, 0.0, 0.0);
cpdf_setlinewidth($pdf, 2);
cpdf_save($pdf);
cpdf_rotate($pdf, -((($time['seconds'] - 15.0) * 6.0));
cpdf_moveto($pdf, -$radius/5, 0.0);
cpdf_lineto($pdf, $radius, 0.0);
cpdf_stroke($pdf);
cpdf_restore($pdf);
/* draw little circle at center */
cpdf_circle($pdf, 0, 0, $radius/30);
cpdf_fill($pdf);
cpdf_restore($pdf);
cpdf_finalize_page($pdf, $pagecount+1);
}
cpdf_finalize($pdf);
Header("Content-type: application/pdf");
cpdf_output_buffer($pdf);
cpdf_close($pdf);
?>

```

[10.9.1 cpdf_global_set_document_limits](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_global_set_document_limits](#)(int *maxpages*, int *maxfonts*, int *maximages*, int *maxannotations*, int *maxobjects*)

[cpdf_global_set_document_limits\(\)](#) permet de fixer plusieurs limites au document PDF. Cette fonction doit être appelé avant [cpdf_open\(\)](#) pour être effective. Elle fixe les limites de tous les documents ouverts après. Voir aussi [cpdf_open\(\)](#).

10.9.2 cpdf_set_creator

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_creator](#)(string *creator*)

[cpdf_set_creator\(\)](#) fixe le créateur d'un document PDF.

Voir aussi [cpdf_set_subject\(\)](#), [cpdf_set_title\(\)](#), [cpdf_set_keywords\(\)](#).

10.9.3 cpdf_set_title

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_title](#)(string *title*)

[cpdf_set_title\(\)](#) fixe le titre d'un document PDF.

Voir aussi [cpdf_set_subject\(\)](#), [cpdf_set_creator\(\)](#), [cpdf_set_keywords\(\)](#).

10.9.4 cpdf_set_subject

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_subject](#)(string *subject*)

[cpdf_set_subject\(\)](#) fixe le sujet d'un document PDF.

Voir aussi [cpdf_set_title\(\)](#), [cpdf_set_creator\(\)](#), [cpdf_set_keywords\(\)](#).

10.9.5 cpdf_set_keywords

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_keywords](#)(string *keywords*)

[cpdf_set_keywords\(\)](#) fixe les mot clés d'un document PDF.

Voir aussi [cpdf_set_title\(\)](#), [cpdf_set_creator\(\)](#), [cpdf_set_subject\(\)](#).

10.9.6 cpdf_open

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [cpdf_open](#)(int *compression*, string *filename*)

[cpdf_open\(\)](#) ouvre un nouveau document PDF. Le premier paramètre active ou pas la compression, suivant qu'il vaut 0 ou 1. Le deuxième paramètre, optionnel, choisit le fichier de destination du document. Si il est omis, le document sera écrit en mémoire, et pourra être écrit dans un fichier avec [cpdf_save_to_file\(\)](#) ou envoyé à l'affichage avec [cpdf_output_buffer\(\)](#). Note : *La valeur retournée sera nécessaire pour les autres fonctions de ClibPDF comme premier paramètre.*

La librairie ClibPDF prend le nom de fichier "-" comme synonyme de stdout. Si PHP est compilé comme un module apache, cela ne fonctionnera pas, car la méthode d'envoi des données de ClibPDF ne fonctionne pas avec Apache. Vous pouvez résoudre ce problème en ne fournissant pas de nom de fichier, et en utilisant la fonction [cpdf_output_buffer\(\)](#) pour afficher le document PDF.

Voir aussi [cpdf_close\(\)](#), [cpdf_output_buffer\(\)](#).

[10.9.7 cpdf_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_close](#)(int *pdf document*)

[cpdf_close\(\)](#) ferme un fichier PDF. Ce doit être la dernière fonction appelée, et elle apparaît même après [cpdf_finalize\(\)](#), [cpdf_output_buffer\(\)](#) et [cpdf_save_to_file\(\)](#).

Voir aussi [cpdf_open\(\)](#).

[10.9.8 cpdf_page_init](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_page_init](#)(int *pdf document*, int *page number*, int *orientation*, double *height*, double *width*, double *unit*)

[cpdf_page_init\(\)](#) commence une nouvelle page, avec la hauteur *height* et la largeur *width*. La page a le numéro *page number* et l'orientation *orientation*. *orientation* vaut 0 pour portrait et 1 pour paysage. Le dernier paramètre, optionnel, *unit*, fixe l'unité pour le système de coordonnées. Cette valeur doit être un nombre de points postscript, par unité. Etant donné que un pouce (inch) vaut 72 points, une valeur de 72 vaudra un pouce (inch). Par défaut, cette valeur vaut 72.

Voir aussi [cpdf_set_current_page\(\)](#).

[10.9.9 cpdf_finalize_page](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_finalize_page](#)(int *pdf document*, int *page number*)

[cpdf_finalize_page\(\)](#) termine la page de numéro *page number*. Cette fonction fait que qu'une sauvegarde mémoire. Les pages terminées prennent moins de place, mais ne peuvent plus être modifiées.

Voir aussi [cpdf_page_init\(\)](#).

[10.9.10 cpdf_finalize](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_finalize](#)(int *pdf document*)

[cpdf_finalize\(\)](#) termine un document. Vous devez toujours appeler [cpdf_close\(\)](#) après.

Voir aussi [cpdf_close\(\)](#).

[10.9.11 cpdf_output_buffer](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_output_buffer](#)(int *pdf document*)

[cpdf_output_buffer\(\)](#) envoie le document PDF dans un buffer mémoire de stdout. Le document doit avoir été

créé en mémoire, ce qui est le cas si [cpdf_open\(\)](#) a été appelée dans paramètre de nom de fichier. Voir aussi [cpdf_open\(\)](#).

10.9.12 cpdf_save_to_file

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_save_to_file](#)(int *pdf document*, string *filename*)

[cpdf_save_to_file\(\)](#) écrit un document PDF dans un fichier, si il a été créé en mémoire. Cette fonction n'est pas nécessaire si un nom de fichier a été fourni lors de l'appel à [cpdf_open\(\)](#). Voir aussi [cpdf_output_buffer\(\)](#), [cpdf_open\(\)](#).

10.9.13 cpdf_set_current_page

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_current_page](#)(int *pdf document*, int *page number*)

[cpdf_set_current_page\(\)](#) fixe la page courante, où toutes les prochaines opérations vont avoir lieu. On peut changer de page jusqu'à ce qu'une page soit terminée avec [cpdf_finalize_page\(\)](#). Voir aussi [cpdf_finalize_page\(\)](#).

10.9.14 cpdf_begin_text

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_begin_text](#)(int *pdf document*)

[cpdf_begin_text\(\)](#) démarre une section de texte. Elle doit être terminée avec [cpdf_end_text\(\)](#). *Affichage de texte*

```
<?php cpdf_begin_text($pdf);
cpdf_set_font($pdf, 16, "Helvetica", "WinAnsiEncoding");
cpdf_text($pdf, 100, 100, "Some text");
cpdf_end_text($pdf) ?>
```

Voir aussi [cpdf_end_text\(\)](#).

10.9.15 cpdf_end_text

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_end_text](#)(int *pdf document*)

[cpdf_end_text\(\)](#) termine une section de texte, commencée avec [cpdf_begin_text\(\)](#). *Affichage de texte*

```
<?php cpdf_begin_text($pdf);
```

```

cpdf_set_font($pdf, 16, "Helvetica", "WinAnsiEncoding");
cpdf_text($pdf, 100, 100, "Some text");
cpdf_end_text($pdf) ?>

```

Voir aussi [cpdf_begin_text\(\)](#).

10.9.16 cpdf_show

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_show](#) (int *pdf document*, string *text*)

[cpdf_show\(\)](#) imprime la chaîne *text*, à la position courante.

Voir aussi [cpdf_text\(\)](#), [cpdf_begin_text\(\)](#), [cpdf_end_text\(\)](#).

10.9.17 cpdf_show_xy

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_show_xy](#) (int *pdf document*, string *text*, double *x-koor*, double *y-koor*, int *mode*)

[cpdf_show_xy\(\)](#) imprime la chaîne *text*, à la position de coordonnées (*x-koor*, *y-koor*). Le dernier paramètre optionnel est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé. Note : *The function [cpdf_show_xy\(\)](#) est identique à [cpdf_text\(\)](#) sans les options.*

Voir aussi [cpdf_text\(\)](#).

10.9.18 cpdf_text

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_text](#) (int *pdf document*, string *text*, double *x-koor*, double *y-koor*, int *mode*, double *orientation*, int *alignmode*)

[cpdf_text\(\)](#) imprime le text *text* à la position de coordonnées (*x-koor*, *y-koor*). Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé.

Le paramètre optionnel *orientation* est un angle de rotation du texte, en degrés. Le paramètre optionnel *alignmode* détermine l'alignement du texte. Reportez vous à la doc de ClibPDF, pour les valeurs possibles.

Voir aussi [cpdf_show_xy\(\)](#).

10.9.19 cpdf_set_font

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_font](#) (int *pdf document*, string *font name*, double *size*, string *encoding*)

[cpdf_set_font\(\)](#) Selectionne la police courante, sa taille, et l'encodage. Actuellement, seules les polices postscript sont supportées. Le dernier paramètre *encoding* peut prendre les valeurs suivantes :

"MacRomanEncoding", "MacExpertEncoding", "WinAnsiEncoding", et "NULL". "NULL" signifie qu'il faut utiliser l'encodage par défaut. Reportez vous à la doc de ClibPDF, pour plus d'informations, notamment sur

les polices asiatiques.

[10.9.20 cpdf_set_leading](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_leading](#)(int *pdf document*, double *distance*)

[cpdf_set_leading\(\)](#) fixe la distance entre deux lignes. Cela servira si le texte est affiché par [cpdf_continue_text\(\)](#).

Voir aussi [cpdf_continue_text\(\)](#).

[10.9.21 cpdf_set_text_rendering](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_text_rendering](#)(int *pdf document*, int *mode*)

[cpdf_set_text_rendering\(\)](#) détermine le rendu du texte. Les valeurs possibles pour *mode* sont : 0=texte plein, 1=texte stroke, 2=texte plein et stroke, 3=invisible, 4=texte plein et ajouté au chemin, 5=texte stroke et ajouté au chemin, 6=texte plein et stroke et ajouté au chemin, 7=et ajouté au chemin.

[10.9.22 cpdf_set_horiz_scaling](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_horiz_scaling](#)(int *pdf document*, double *scale*)

[cpdf_set_horiz_scaling\(\)](#) fixe l'échelle horizontale du texte à *scale* %.

[10.9.23 cpdf_set_text_rise](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_text_rise](#)(int *pdf document*, double *value*)

[cpdf_set_text_rise\(\)](#) fixe l'élévation du texte à *value* unités.

[10.9.24 cpdf_set_text_matrix](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_text_matrix](#)(int *pdf document*, array *matrix*)

[cpdf_set_text_matrix\(\)](#) fixe la matrice du texte, qui décrit la transformation appliquée à police.

[10.9.25 cpdf_set_text_pos](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_text_pos](#)(int *pdf document*, double *x-koor*, double *y-koor*, int *mode*)

[cpdf_set_text_pos\(\)](#) Fixe la position du texte pour le prochain appel à [cpdf_show\(\)](#).

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé.

Voir aussi [cpdf_show\(\)](#), [cpdf_text\(\)](#).

10.9.26 cpdf_set_char_spacing

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_char_spacing](#)(int *pdf document*, double *space*)

[cpdf_set_char_spacing\(\)](#) fixe l'espacement des caractères.

Voir aussi [cpdf_set_word_spacing\(\)](#), [cpdf_set_leading\(\)](#).

10.9.27 cpdf_set_word_spacing

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_word_spacing](#)(int *pdf document*, double *space*)

[cpdf_set_word_spacing\(\)](#) fixe l'espacement des caractères.

Voir aussi [cpdf_set_char_spacing\(\)](#), [cpdf_set_leading\(\)](#).

10.9.28 cpdf_continue_text

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_continue_text](#)(int *pdf document*, string *text*)

[cpdf_continue_text\(\)](#) imprime le texte *text* à la ligne suivante.

Voir aussi [cpdf_show_xy\(\)](#), [cpdf_text\(\)](#), [cpdf_set_leading\(\)](#), [cpdf_set_text_pos\(\)](#).

10.9.29 cpdf_stringwidth

[\[Notes en ligne\]](#) [\[Exemples\]](#)

double [cpdf_stringwidth](#)(int *pdf document*, string *text*)

[cpdf_stringwidth\(\)](#) retourne la taille de la chaîne *text*. Une police doit avoir déjà été choisie.

Voir aussi [cpdf_set_font\(\)](#).

10.9.30 cpdf_save

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_save](#)(int *pdf document*)

[cpdf_save\(\)](#) sauve l'environnement courant. Cette fonction est similaire à la commande postscript gsave. Très pratique quand vous devez faire des translations et rotations sur un objet, mais sans affecter les autres.

Voir aussi [cpdf_restore\(\)](#).

10.9.31 [cpdf_restore](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_restore](#) (int *pdf document*)

[cpdf_restore\(\)](#) restaure l'environnement sauvé par [cpdf_save\(\)](#). Cette fonction est similaire à la commande postscript grestore. Très pratique quand vous devez faire des translations et rotations sur un objet, mais sans affecter les autres. *Sauver/Restaurer*

```
<?php cpdf_save($pdf);
// plein de transformations, translations, ...
cpdf_restore($pdf) ?>
```

Voir aussi [cpdf_save\(\)](#).

10.9.32 [cpdf_translate](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_translate](#) (int *pdf document*, double *x-koor*, double *y-koor*, int *mode*)

[cpdf_translate\(\)](#) modifie l'origine du système de coordonnées en plaçant l'origine aux coordonnées (*x-koor*, *y-koor*).

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé.

10.9.33 [cpdf_scale](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_scale](#) (int *pdf document*, double *x-scale*, double *y-scale*)

[cpdf_scale\(\)](#) modifie l'échelle dans les deux direction.

10.9.34 [cpdf_rotate](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_rotate](#) (int *pdf document*, double *angle*)

[cpdf_rotate\(\)](#) effectue une rotation, d'un angle de *angle* degrés.

10.9.35 [cpdf_setflat](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setflat](#) (int *pdf document*, double *value*)

[cpdf_setflat\(\)](#) fixe la platitude (flatness), entre 0 et 100.

[10.9.36 cpdf_setlinejoin](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setlinejoin](#)(int *pdf document*, long *value*)

[cpdf_setlinejoin\(\)](#) fixe le paramètre linejoin à une valeur entre 0 et 2. 0 = miter, 1 = round, 2 = bevel.

[10.9.37 cpdf_setlinecap](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setlinecap](#)(int *pdf document*, int *value*)

[cpdf_setlinecap\(\)](#) fixe le paramètre linecap à une valeur entre 0 et 2. 0 = butt end, 1 = round, 2 = projecting square.

[10.9.38 cpdf_setmiterlimit](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setmiterlimit](#)(int *pdf document*, double *value*)

[cpdf_setmiterlimit\(\)](#) Fixe le paramètre miter limit à une valeur supérieure ou égale à 1.

[10.9.39 cpdf_setlinewidth](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setlinewidth](#)(int *pdf document*, double *width*)

[cpdf_setlinewidth\(\)](#) Fixe la largeur de ligne à la valeur de *width*.

[10.9.40 cpdf_setdash](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setdash](#)(int *pdf document*, double *white*, double *black*)

[cpdf_setdash\(\)](#) fixe le motif de pointillé à *white* unité de blanc et *black* unités de noir. Si les deux sont à 0, une ligne pleine est affichée.

[10.9.41 cpdf_moveto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_moveto](#)(int *pdf document*, double *x-koor*, double *y-koor*, int *mode*)

[cpdf_moveto\(\)](#) fixe le point courant aux coordonnées (*x-koor*, *y-koor*).

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisée.

10.9.42 cpdf_rmoveto

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_rmoveto](#)(int *pdf document*, double *x-koor*, double *y-koor*, int *mode*)

[cpdf_rmoveto\(\)](#) Fixe le point courant aux coordonnées (*x-koor*, *y-koor*), relativement.

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisée.

Voir aussi [cpdf_moveto\(\)](#).

10.9.43 cpdf_curveto

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_curveto](#)(int *pdf document*, double *x1*, double *y1*, double *x2*, double *y2*, double *x3*, double *y3*, int *mode*)

[cpdf_curveto\(\)](#) dessine une courbe de Bezier, entre le point courant et le point (*x3*, *y3*), en utilisant les points de contrôle (*x1*, *y1*) et (*x2*, *y2*).

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisée.

Voir aussi [cpdf_moveto\(\)](#), [cpdf_rmoveto\(\)](#), [cpdf_rlineto\(\)](#), [cpdf_lineto\(\)](#).

10.9.44 cpdf_lineto

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_lineto](#)(int *pdf document*, double *x-koor*, double *y-koor*, int *mode*)

[cpdf_lineto\(\)](#) dessine une ligne entre le point courant et le point de coordonnées (*x-koor*, *y-koor*).

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé.

Voir aussi [cpdf_moveto\(\)](#), [cpdf_rmoveto\(\)](#), [cpdf_curveto\(\)](#).

10.9.45 cpdf_rlineto

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_rlineto](#)(int *pdf document*, double *x-koor*, double *y-koor*, int *mode*)

[cpdf_rlineto\(\)](#) dessine une ligne entre le point courant et le point de coordonnées relatives (*x-koor*, *y-koor*).

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé.

Voir aussi [cpdf_moveto\(\)](#), [cpdf_rmoveto\(\)](#), [cpdf_curveto\(\)](#).

10.9.46 cpdf_circle

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_circle](#)(int *pdf document*, double *x-koor*, double *y-koor*, double *radius*,

int *mode*)

[cpdf_circle\(\)](#) dessine un cercle de centre (*x-koor*, *y-koor*) et de rayon *radius*.

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé.

Voir aussi [cpdf_arc\(\)](#).

[10.9.47 cpdf_arc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_arc](#)(int *pdf document*, double *x-koor*, double *y-koor*, double *radius*, double *start*, double *end*, int *mode*)

[cpdf_arc\(\)](#) Dessine un arc de cercle, dont le centre est au point (*x-koor*, *y-koor*) et l'angle est *radius*, commençant à l'angle *start* et finissant à l'angle *end*.

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé.

Voir aussi [cpdf_circle\(\)](#).

[10.9.48 cpdf_rect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_rect](#)(int *pdf document*, double *x-koor*, double *y-koor*, double *width*, double *height*, int *mode*)

[cpdf_rect\(\)](#) Dessine un rectangle dont le coin inférieur droit est au point (*x-koor*, *y-koor*). La largeur est *width*. La hauteur est *height*.

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé.

[10.9.49 cpdf_closepath](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_closepath](#)(int *pdf document*)

[cpdf_closepath\(\)](#) ferme le chemin courant.

[10.9.50 cpdf_stroke](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_stroke](#)(int *pdf document*)

[cpdf_stroke\(\)](#) dessine une ligne le long du chemin.

Voir aussi [cpdf_closepath\(\)](#), [cpdf_closepath_stroke\(\)](#).

[10.9.51 cpdf_closepath_stroke](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_closepath_stroke](#)(int *pdf document*)

[cpdf_closepath_stroke\(\)](#) est une combinaison de [cpdf_closepath\(\)](#) et [cpdf_stroke\(\)](#).
Voir aussi [cpdf_closepath\(\)](#), [cpdf_stroke\(\)](#).

[10.9.52 cpdf_fill](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_fill](#)(int *pdf document*)

[cpdf_fill\(\)](#) remplit l'intérieur du chemin courant avec la couleur courante.
Voir aussi [cpdf_closepath\(\)](#), [cpdf_stroke\(\)](#), [cpdf_setgray_fill\(\)](#), [cpdf_setgray\(\)](#), [cpdf_setrgbcolor_fill\(\)](#), [cpdf_setrgbcolor\(\)](#).

[10.9.53 cpdf_fill_stroke](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_fill_stroke](#)(int *pdf document*)

[cpdf_fill_stroke\(\)](#) remplit l'intérieur du chemin avec la couleur courante, et dessine le chemin.
Voir aussi [cpdf_closepath\(\)](#), [cpdf_stroke\(\)](#), [cpdf_fill\(\)](#), [cpdf_setgray_fill\(\)](#), [cpdf_setgray\(\)](#), [cpdf_setrgbcolor_fill\(\)](#), [cpdf_setrgbcolor\(\)](#).

[10.9.54 cpdf_closepath_fill_stroke](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_closepath_fill_stroke](#)(int *pdf document*)

[cpdf_closepath_fill_stroke\(\)](#) remplit le chemin, dessine le bord et ferme le chemin.
Voir aussi [cpdf_closepath\(\)](#), [cpdf_stroke\(\)](#), [cpdf_fill\(\)](#), [cpdf_setgray_fill\(\)](#), [cpdf_setgray\(\)](#), [cpdf_setrgbcolor_fill\(\)](#), [cpdf_setrgbcolor\(\)](#).

[10.9.55 cpdf_clip](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_clip](#)(int *pdf document*)

[cpdf_clip\(\)](#) aligne les dessins sur le chemin courant.

[10.9.56 cpdf_setgray_fill](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setgray_fill](#)(int *pdf document*, double *value*)

[cpdf_setgray_fill\(\)](#) Choisit un niveau de gris comme couleur de remplissage.
Voir aussi [cpdf_setrgbcolor_fill\(\)](#).

10.9.57 cpdf_setgray_stroke

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setgray_stroke](#)(int *pdf document*, double *gray value*)

[cpdf_setgray_stroke\(\)](#) choisit un niveau de gris comme couleur de dessin.
Voir aussi [cpdf_setrgbcolor_stroke\(\)](#).

10.9.58 cpdf_setgray

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setgray](#)(int *pdf document*, double *gray value*)

[cpdf_setgray_stroke\(\)](#) choisit un niveau de gris comme couleur de dessin et de remplissage.
Voir aussi [cpdf_setrgbcolor_stroke\(\)](#), [cpdf_setrgbcolor_fill\(\)](#).

10.9.59 cpdf_setrgbcolor_fill

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setrgbcolor_fill](#)(int *pdf document*, double *red value*, double *green value*, double *blue value*)

[cpdf_setrgbcolor_fill\(\)](#) choisit une couleur rgb comme couleur de remplissage.
Voir aussi [cpdf_setrgbcolor_stroke\(\)](#), [cpdf_setrgbcolor\(\)](#).

10.9.60 cpdf_setrgbcolor_stroke

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setrgbcolor_stroke](#)(int *pdf document*, double *red value*, double *green value*, double *blue value*)

[cpdf_setrgbcolor_stroke\(\)](#) choisit une couleur rgb comme couleur de dessin.
Voir aussi [cpdf_setrgbcolor_fill\(\)](#), [cpdf_setrgbcolor\(\)](#).

10.9.61 cpdf_setrgbcolor

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setrgbcolor](#)(int *pdf document*, double *red value*, double *green value*, double *blue value*)

[cpdf_setrgbcolor_stroke\(\)](#) choisit une couleur rgb comme couleur de dessin et de remplissage.
Voir aussi [cpdf_setrgbcolor_stroke\(\)](#), [cpdf_setrgbcolor_fill\(\)](#).

[10.9.62 cpdf_add_outline](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_add_outline](#)(int *pdf document*, string *text*)

[cpdf_add_outline\(\)](#) ajoute un signet à la page courante, avec le texte *text* qui pointe sur la page courante.

Ajouter une mise en relief

```

<?php
$cpdf = cpdf_open(0);
cpdf_page_init($cpdf, 1, 0, 595, 842);
cpdf_add_outline($cpdf, 0, 0, 0, 1, "Page 1");
// ...
// quelques dessin
// ...
cpdf_finalize($cpdf);
Header("Content-type: application/pdf");
cpdf_output_buffer($cpdf);
cpdf_close($cpdf);
?>

```

[10.9.63 cpdf_set_page_animation](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_page_animation](#)(int *pdf document*, int *transition*, double *duration*)

[cpdf_set_page_animation\(\)](#) Fixe l'animation de la transition entre les pages.

La valeur du paramètre de transition *transition* peut être

- 0 pour aucune,
- 1 pour deux lignes en travers de l'écran, qui révèlent la prochaine page,
- 2 pour plusieurs lignes en travers de l'écran, qui révèlent la prochaine page,
- 3 pour une boîte qui révèle la prochaine page,
- 4 pour une seule ligne en travers de l'écran, qui révèle la prochaine page,
- 5 pour l'ancienne page qui se dissout
- 6 pour un effet de dissolution d'un angle à l'autre
- 7 pour le remplacement simple (par défaut)

La valeur de *duration* est le nombre de secondes avant le passage à la page suivante.

[10.9.64 cpdf_import_jpeg](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [cpdf_import_jpeg](#)(int *pdf document*, string *file name*, double *x-koor*, double *y-koor*, double *angle*, double *width*, double *height*, double *x-scale*, double *y-scale*,

int *mode*)

[cpdf_import_jpeg\(\)](#) ouvre une image JPG, enregistré dans le fichier *file name*. Le format de l'image doit être JPEG. L'image est placée dans la page courante, aux coordonnées (*x-koor*, *y-koor*). L'image subira une rotation d'un angle de *angle* degrés.

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisée.

Voir aussi [cpdf_place_inline_image\(\)](#).

[10.9.65 cpdf_place_inline_image](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_place_inline_image](#)(int *pdf document*, int *image*, double *x-koor*, double *y-koor*, double *angle*, double *width*, double *height*, int *mode*)

[cpdf_place_inline_image\(\)](#) places une image créée par un script PHP, dans la page, à la position (*x-koor*, *y-koor*). L'image peut être mise à l'échelle, en même temps.

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé.

Voir aussi [cpdf_import_jpeg\(\)](#).

[10.9.66 cpdf_add_annotation](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_add_annotation](#)(int *pdf document*, double *llx*, double *lly*, double *urx*, double *ury*, string *title*, string *content*, int *mode*)

[cpdf_add_annotation\(\)](#) ajoute une note, dont le coin inférieur droit est (*llx*, *lly*) et le coin supérieur droit est (*urx*, *ury*).

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé.

[10.10 CURL, Client URL Library Functions](#)

[\[Notes en ligne\]](#)

PHP supports libcurl, a library, created by Daniel Stenberg, that allows you to connect and communicate to many different types of servers with many different types of protocols. libcurl currently supports the http, https, ftp, gopher, telnet, dict, file, and ldap protocols. libcurl also supports HTTPS certificates, HTTP POST, HTTP PUT, FTP uploading (this can also be done with PHP's ftp extension), HTTP form based upload, proxies, cookies and user+password authentication.

In order to use the CURL functions you need to install the [CURL](#) package. PHP requires that you use CURL 7.0.2-beta or higher. PHP will not work with any version of CURL below version 7.0.2-beta.

To use PHP's CURL support you must also compile PHP `--with-curl[=DIR]` where DIR is the location of the directory containing the lib and include directories. In the "include" directory there should be a folder named "curl" which should contain the easy.h and curl.h files. There should be a file named "libcurl.a" located in the "lib" directory.

Once you've compiled PHP with CURL support, you can begin using the curl functions. The basic idea

behind the CURL functions is that you initialize a CURL session using the [curl_init\(\)](#), then you can set all your options for the transfer via the [curl_exec\(\)](#) and then you finish off your session using the [curl_close\(\)](#). Here is an example that uses the CURL functions to fetch the PHP homepage into a file: *Using PHP's CURL module to fetch the PHP homepage*

```
<?php
$ch = curl_init ("http://www.php.net/");
$fp = fopen ("php_homepage.txt", "w");
curl_setopt ($ch, CURLOPT_INFILE, $fp);
curl_setopt ($ch, CURLOPT_HEADER, 0);
curl_exec ($ch);
curl_close ($ch);
fclose ($fp);
?>
```

[10.10.1 curl_init](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [curl_init](#)(string *url*)

The [curl_init\(\)](#) will initialize a new session and return a CURL handle for use with the [curl_setopt\(\)](#), [curl_exec\(\)](#), and [curl_close\(\)](#) functions. If the optional *url* parameter is supplied then the CURLOPT_URL option will be set to the value of the parameter. You can manually set this using the [curl_setopt\(\)](#) function.

Initializing a new CURL session and fetching a webpage *Initializing a new CURL session and fetching a webpage*

```
<?php
$ch = curl_init();
curl_setopt ($ch, CURLOPT_URL, "http://www.zend.com/");
curl_setopt ($ch, CURLOPT_HEADER, 0);
curl_exec ($ch);
curl_close ($ch);
?>
```

See also: [curl_close\(\)](#), [curl_setopt\(\)](#)

[10.10.2 curl_setopt](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [curl_setopt](#)(int *ch* , string *option* , mixed *value*)

The [curl_setopt\(\)](#) function will set options for a CURL session identified by the *ch* parameter. The *option* parameter is the option you want to set, and the *value* is the value of the option given by the *option*. The *value* should be a long for the following options (specified in the *option* parameter):

- ***CURLOPT_INFILESIZE***: When you are uploading a file to a remote site, this option should be used to tell PHP what the expected size of the infile will be.
- ***CURLOPT_VERBOSE***: Set this option to a non-zero value if you want CURL to report everything that is happening.
- ***CURLOPT_HEADER***: Set this option to a non-zero value if you want the header to be included in the output.
- ***CURLOPT_NOPROGRESS***: Set this option to a non-zero value if you don't want PHP to display a progress meter for CURL transfers Note : *PHP automatically sets this option to a non-zero parameter, this should only be changed for debugging purposes.*
- ***CURLOPT_NOBODY***: Set this option to a non-zero value if you don't want the body included with the output.
- ***CURLOPT_FAILONERROR***: Set this option to a non-zero value if you want PHP to fail silently if the HTTP code returned is greater than 300. The default behaviour is to return the page normally, ignoring the code.
- ***CURLOPT_UPLOAD***: Set this option to a non-zero value if you want PHP to prepare for an upload.
- ***CURLOPT_POST***: Set this option to a non-zero value if you want PHP to do a regular HTTP POST. This POST is a normal application/x-www-form-urlencoded kind, most commonly used by HTML forms.
- ***CURLOPT_FTPLISTONLY***: Set this option to a non-zero value and PHP will just list the names of an FTP directory.
- ***CURLOPT_FTPAPPEND***: Set this option to a non-zero value and PHP will append to the remote file instead of overwriting it.
- ***CURLOPT_NETRC***: Set this option to a non-zero value and PHP will scan your ~/.netrc file to find your username and password for the remote site that you're establishing a connection with.
- ***CURLOPT_FOLLOWLOCATION***: Set this option to a non-zero value to follow any "Location: " header that the server sends as a part of the HTTP header (note this is recursive, PHP will follow as many "Location: " headers that it is sent.)
- ***CURLOPT_PUT***: Set this option a non-zero value to HTTP PUT a file. The file to PUT must be set with the CURLOPT_INFILE and CURLOPT_INFILESIZE.
- ***CURLOPT_MUTE***: Set this option to a non-zero value and PHP will be completely silent with regards to the CURL functions.
- ***CURLOPT_TIMEOUT***: Pass a long as a parameter that contains the maximum time, in seconds, that you'll allow the curl functions to take.
- ***CURLOPT_LOW_SPEED_LIMIT***: Pass a long as a parameter that contains the transfer speed in bytes per second that the transfer should be below during CURLOPT_LOW_SPEED_TIME seconds for PHP to consider it too slow and abort.
- ***CURLOPT_LOW_SPEED_TIME***: Pass a long as a parameter that contains the time in seconds that the transfer should be below the CURLOPT_LOW_SPEED_LIMIT for PHP to consider it too slow and abort.
- ***CURLOPT_RESUME_FROM***: Pass a long as a parameter that contains the offset, in bytes, that you want the transfer to start from.
- ***CURLOPT_SSLVERSION***: Pass a long as a parameter that contains the SSL version (2 or 3) to use. By default PHP will try and determine this by itself, although, in some cases you must set this manually.
- ***CURLOPT_TIMECONDITION***: Pass a long as a parameter that defines how the CURLOPT_TIMEVALUE is treated. You can set this parameter to TIMECOND_IFMODSINCE or TIMECOND_ISUNMODSINCE. This is a HTTP-only feature.
- ***CURLOPT_TIMEVALUE***: Pass a long as a parameter that is the time in seconds since January 1st, 1970. The time will be used as specified by the CURLOPT_TIMEVALUE option, or by default the TIMECOND_IFMODSINCE will be used.

The *value* parameter should be a string for the following values of the *option* parameter:

- ***CURLOPT_URL***: This is the URL that you want PHP to fetch. You can also set this option when initializing a session with the [curl_init\(\)](#) function.
- ***CURLOPT_USERPWD***: Pass a string formatted in the [username]:[password] manner, for PHP to use for the connection.
- ***CURLOPT_PROXYUSERPWD***: Pass a string formatted in the [username]:[password] format for connection to the HTTP proxy.
- ***CURLOPT_RANGE***: Pass the specified range you want. It should be in the "X-Y" format, where X or Y may be left out. The HTTP transfers also support several intervals, separated with commas as in X-Y,N-M.
- ***CURLOPT_POSTFIELDS***: Pass a string containing the full data to post in an HTTP "POST" operation.
- ***CURLOPT_REFERER***: Pass a string containing the "referer" header to be used in an HTTP request.
- ***CURLOPT_USERAGENT***: Pass a string containing the "user-agent" header to be used in an HTTP request.
- ***CURLOPT_FTPPORT***: Pass a string containing the which will be used to get the IP address to use for the ftp "PORT" instruction. The POST instruction tells the remote server to connect to our specified IP address. The string may be a plain IP address, a hostname, a network interface name (under UNIX), or just a plain '-' to use the systems default IP address.
- ***CURLOPT_COOKIE***: Pass a string containing the content of the cookie to be set in the HTTP header.
- ***CURLOPT_SSLCERT***: Pass a string containing the filename of PEM formatted certificate.
- ***CURLOPT_SSLCERTPASSWD***: Pass a string containing the password required to use the CURLOPT_SSLCERT certificate.
- ***CURLOPT_COOKIEFILE***: Pass a string containing the name of the file containing the cookie data. The cookie file can be in Netscape format, or just plain HTTP-style headers dumped into a file.
- ***CURLOPT_CUSTOMREQUEST***: Pass a string to be used instead of GET or HEAD when doing an HTTP request. This is useful for doing DELETE or another, more obscure, HTTP request. Note : *Don't do this without making sure your server supports the command first.*

The following options expect a file descriptor that is obtained by using the [fopen\(\)](#) function:

- ***CURLOPT_FILE***: The file where the output of your transfer should be placed, the default is STDOUT.
- ***CURLOPT_INFILE***: The file where the input of your transfer comes from.
- ***CURLOPT_WRITEHEADER***: The file to write the header part of the output into.
- ***CURLOPT_STDERR***: The file to write errors to instead of stderr.

[10.10.3 curl_exec](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

```
bool curl\_exec(int ch )
```

This function is should be called after you initialize a CURL session and all the options for the session are

set. Its purpose is simply to execute the predefined CURL session (given by the *ch*).

[10.10.4 curl_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [curl_close](#) (int *ch*)

This functions closes a CURL session and frees all ressources. The CURL handle, *ch*, is also deleted.

[10.10.5 curl_version](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [curl_version](#)

The [curl_version\(\)](#) function returns a string containing the current CURL version.

[10.11 Fonctions de paiement Cybercash](#)

[\[Notes en ligne\]](#)

Ces fonctions ne sont disponibles que si PHP a été compilé avec l'option `--with-cybercash=[DIR]`. Ces fonctions ont été ajoutées dans PHP4.

[10.11.1 cybercash_encr](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [cybercash_encr](#) (string *wmk*, string *sk*, string *inbuff*)

Cette fonction retourne un tableau associatif, contenant les éléments "errcode" et, si "errcode" vaut FALSE, "outbuff" (string), "outLth" (long) et "macbuff" (string).

[10.11.2 cybercash_decr](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [cybercash_decr](#) (string *wmk*, string *sk*, string *inbuff*)

Cette fonction retourne un tableau associatif, contenant les éléments "errcode" et, si "errcode" vaut FALSE, "outbuff" (string), "outLth" (long) et "macbuff" (string).

[10.11.3 cybercash_base64_encode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [cybercash_base64_encode](#) (string *inbuff*)

10.11.4 cybercash_base64_decode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [cybercash_base64_decode](#) (string *inbuff*)

10.12 Fonctions de dates et heures

[\[Notes en ligne\]](#)

10.12.1 checkdate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [checkdate](#) (int *month*, int *day*, int *year*)

Retourne TRUE si la date fournie est valide, et sinon FALSE. La date est considérée comme valide si :

- L'année est comprise entre 0 et 32767 inclus
- Le mois est compris entre 1 et 12 inclus
- Le jour est compris dans l'intervalle de date du mois. Les années bissextiles sont prises en compte.

10.12.2 date

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [date](#) (string *format*, int *timestamp*)

Retourne une date sous forme d'une chaîne, au format donné par la chaîne format. La date est fournie sous la forme d'un *timestamp*. Par défaut, la date courante est utilisée.

Les caractères suivants sont utilisés pour spécifier le format :

- a – "am" (matin) ou "pm" (après-midi)
- A – "AM" (matin) ou "PM" (après-midi)
- d – Jour du mois, sur deux chiffres (éventuellement avec un zéro) : "01" à "31"
- D – Jour de la semaine, en trois lettres (et en anglais) : par exemple "Fri" (pour Vendredi)
- F – Mois, textuel, version longue; en anglais, i.e. "January" (pour Janvier)
- h – Heure, au format 12h, "01" à "12"
- H – heure, au format 24h, "00" à "23"
- g – Heure, au format 12h sans les zéros initiaux, "1" à "12"
- G – Heure, au format 24h sans les zéros initiaux, "0" à "23"
- i – Minutes; "00" à "59"
- j – Jour du mois sans les zéros initiaux: "1" à "31"
- l – ('L' minuscule) – Jour de la semaine, textuel, version longue; en anglais, i.e. "Friday" (pour Vendredi)
- L – Booléen pour savoir si l'année est bisextile ("1") ou pas ("0")
- m – Mois; i.e. "01" à "12"

- n – Mois sans les zéros initiaux; i.e. "1" à "12"
- M – Mois, en trois lettres (et en anglais) : par exemple "Jan" (pour Janvier)
- s – Secondes; i.e. "00" à "59"
- S – Suffixe ordinal d'un nombre, en anglais, sur deux lettres : i.e. "th", "nd"
- t – Nombre de jour dans le mois donné, i.e. "28" à "31"
- U – Secondes depuis une époque
- w – Jour de la semaine, numérique, i.e. "0" (Dimanche) to "6" (Samedi)
- Y – Année, 4 chiffres; i.e. "1999"
- y – Année, 2 chiffres; i.e. "99"
- z – Jour de l'année; i.e. "0" à "365"
- Z – Décalage horaire en secondes (i.e. "-43200" à "43200")

Les caractères non reconnus seront imprimés tels quel. "Z" retournera toujours "0" lorsqu'il est utilisé avec [gmdate\(\)](#). *Exemple avec [date\(\)](#)*

```
print (date("l dS of F Y h:i:s A"));
print ("July 1, 2000 is on a " . date("l", mktime(0,0,0,7,1,2000)));
```

Il est possible d'utiliser [date\(\)](#) et [mktime\(\)](#) ensemble pour générer des dates dans futur ou dans passé. exemples avec [date\(\)](#) et [mktime\(\)](#) *exemples avec [date\(\)](#) et [mktime\(\)](#)*

```
$tomorrow = mktime (0,0,0,date("m") ,date("d")+1,date("Y"));
$lastmonth = mktime (0,0,0,date("m")-1,date("d"), date("Y"));
$nextyear = mktime (0,0,0,date("m"), date("d"), date("Y")+1);
```

Pour formater des dates dans d'autres langues, utilisez les fonctions [setlocale\(\)](#) et [strftime\(\)](#). Voir aussi [gmdate\(\)](#) et [mktime\(\)](#).

10.12.3 [getdate](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [getdate](#) (int *timestamp*)

Retourne un tableau associatif contenant les informations de date et heure du timestamp, avec les champs suivants :

- "seconds" – secondes
- "minutes" – minutes
- "hours" – heures
- "mday" – jour du mois
- "wday" – jour de la semaine, numérique
- "mon" – mois, numérique
- "year" – année, numérique
- "yday" – jour de l'année, numérique; i.e. "299"
- "weekday" – jour de la semaine, texte complet (en anglais); i.e. "Friday"

- "month" – mois, texte complet (en anglais); i.e. "January"

[10.12.4 gettimeofday](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [gettimeofday](#)(void)

C'est une interface vers `gettimeofday(2)`. Elle retourne un tableau associatif qui contient les informations retournées par le système :

- "sec" – secondes
- "usec" – microsecondes
- "minuteswest" – minutes de décalage par rapport à Greenwich, vers l'Ouest.
- "dstime" – type de correction dst

[10.12.5 gmdate](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gmdate](#)(string *format*, int *timestamp*)

Identique à la fonction [date\(\)](#), seulement le temps retourné est GMT (Greenwich Mean Time) Par exemple, en Finlande (GMT +0200), la première ligne ci-dessous affiche "Jan 01 1998 00:00:00", tandis que la seconde "Dec 31 1997 22:00:00". *Exemple avec [gmdate\(\)](#)*

```
echo date ("M d Y H:i:s", mktime (0,0,0,1,1,1998));
echo gmdate ("M d Y H:i:s", mktime (0,0,0,1,1,1998));
```

Voir aussi [date\(\)](#), [mktime\(\)](#), et [gmmktime\(\)](#).

[10.12.6 gmmktime](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gmmktime](#)(int *hour*, int *minute*, int *second*, int *month*, int *day*, int *year*, int *is_dst*)

Identique à [mktime\(\)](#) hormis le fait que les paramètres passés sont GMT.

[10.12.7 gmstrftime](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gmstrftime](#)(string *format*, int *timestamp*)

[gmstrftime\(\)](#) se comporte exactement comme [strftime\(\)](#) hormis le fait l'heure utilisée est celle de Greenwich (Greenwich Mean Time (GMT)). Par exemple, dans la zone Eastern Standard Time (est des USA) (GMT -0500), la première ligne de l'exemple ci dessous affiche "Dec 31 1998 20:00:00", tandis que la seconde affiche "Jan 01 1999 01:00:00". *Exemple avec [gmstrftime\(\)](#)*

```
setlocale ('LC_TIME', 'en_US');
echo strftime ("%b %d %Y %H:%M:%S", mktime (20,0,0,12,31,98))."\n";
echo gmstrftime ("%b %d %Y %H:%M:%S", mktime (20,0,0,12,31,98))."\n";
```

Voir aussi [strftime\(\)](#).

10.12.8 microtime

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [microtime](#)

Retourne la chaîne "msec sec" avec sec qui est mesurée en secondes depuis le début de l'époque UNIX, (1er janvier 1970 00:00:00 GMT), et msec qui est le nombre de microsecondes de cette heure. Cette fonction est seulement disponible sur les systèmes d'exploitation qui supportent la fonction [gettimeofday\(\)](#).

Voir aussi [time\(\)](#).

10.12.9 mktime

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mktime](#) (int *hour*, int *minute*, int *second*, int *month*, int *day*, int *year*, int *is_dst*)

ATTENTION : l'ordre des arguments est différent de celui de la commande UNIX habituelle [mktime\(\)](#), et fournit des résultats aléatoires si on oublie cet ordre. C'est une erreur très commune que de se tromper de sens.

Cette fonction retourne un timestamp UNIX correspondant aux arguments fournis. Ce timestamp est un entier long, contenant le nombre de secondes entre le début de l'époque UNIX (1er Janvier 1970) et le temps spécifié.

Les arguments peuvent être omis, de gauche à droite, et tous les arguments manquants sont utilisés avec la valeur courante de l'heure et du jour.

is_dst peut être mis à 1 si l'heure d'hiver est appliquée, 0 si elle ne l'est pas, et -1 (par défaut) si on ne sait pas.

Note : *is_dst* a été ajouté à partir de la version 3.0.10.

[mktime\(\)](#) est pratique pour faire des calculs de dates et des validations, car elle va automatiquement corriger les valeurs invalides. Par exemple, toutes les lignes suivantes vont retourner la même date : "Jan-01-1998".

Exemple [mktime\(\)](#)

```
echo date ("M-d-Y", mktime (0,0,0,12,32,1997));
echo date ("M-d-Y", mktime (0,0,0,13,1,1997));
echo date ("M-d-Y", mktime (0,0,0,1,1,1998));
echo date ("M-d-Y", mktime (0,0,0,1,1,98));
```

year peut prendre deux ou quatre chiffres, avec les valeurs entre 0–69 qui correspondent à 2000–2069 et 70–99 à 1970–1999 (sur les systèmes où `time_t` sont sur des entiers 32bit signés, comme cela se fait le plus souvent de nos jours, **year** est valide dans l'intervalle 1902 and 2037).

Le dernier jours d'un mois peut être décrit comme le jour "0" du mois suivant, et non pas le jour –1. Les deux exemples suivants vont donner : "Le dernier jour de Février 2000 est: 29". *Last day of next month*

```
$lastday = mktime (0,0,0,3,0,2000);
echo strftime ("Le dernier jour de Février 2000 est: %d", $lastday);
$lastday = mktime (0,0,0,4,-31,2000);
echo strftime ("Le dernier jour de Février 2000 est: %d", $lastday);
```

Voir aussi [date\(\)](#) et [time\(\)](#).

10.12.10 strftime

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strftime](#) (string *format*, int *timestamp*)

Retourne la date sous la forme d'une chaîne formatée conformément au format *format*, en utilisant le timestamp *timestamp* donné. Si le *timestamp* est omis, la date actuelle est utilisée. Les mois et jours de la semaine, et toutes les chaînes dépendantes de la langue sont fixées avec la commande [setlocale\(\)](#).

Les caractères suivant sont utilisés pour spécifier le format de la date :

- %a – nom abrégé du jour de la semaine (local).
- %A – nom complet du jour de la semaine (local).
- %b – nom abrégé du mois (local).
- %B – nom complet du mois (local).
- %c – représentation préférée pour les dates et heures, en local. locale
- %d – jour du mois en numérique (intervalle 00 à 31)
- %H – heure de la journée en numérique, et sur 24–heures (intervalle 00 à 23)
- %I – heure de la journée en numérique, et sur 12– heures (intervalle 01 à 12)
- %j – jour de l'année, en numérique (intervalle 001 à 366)
- %m – mois en numérique (intervalle 1 à 12)
- %M – minute en numérique
- %p – soit `am' ou `pm' en fonction de l'heure absolue, ou en fonction des valeurs enregistrées en local.
- %S – secondes en numérique
- %U – numéro de semaine dans l'année, en considérant le premier dimanche de l'année comme le premier jour de la première semaine.
- %W – numéro de semaine dans l'année, en considérant le premier lundi de l'année comme le premier jour de la première semaine
- %w – jour de la semaine, numérique, avec Dimanche = 0
- %x – format préféré de représentation de la date sans l'heure
- %X – format préféré de représentation de l'heure sans la date
- %y – l'année, numérique, sur deux chiffres (de 00 à 99)
- %Y – l'année, numérique, sur quatre chiffres
- %Z – fuseau horaire, ou nom ou abréviation
- %% – un caractère `%' littéral

Exemple [strftime\(\)](#)

```
setlocale ("LC_TIME", "C");
print(strftime("%A en Finlandais est "));
setlocale ("LC_TIME", "fi");
print(strftime("%A, en Français "));
setlocale ("LC_TIME", "fr");
print(strftime("%A est en Allemand "));
setlocale ("LC_TIME", "de");
print(strftime("%A.\n"));
```

Cet exemple ne fonctionnera que si vous avez les configurations respectives installées sur votre système. Voir aussi [setlocale\(\)](#) et [mktime\(\)](#).

[10.12.11 time](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [time](#)

Retourne la heure courante, mesurée en secondes depuis le début de l'époque UNIX, (1er janvier 1970 00:00:00 GMT).

Voir aussi [date\(\)](#).

[10.13 Fonctions dbm](#)

[\[Notes en ligne\]](#)

Ces fonctions sont l'interface avec les bases de type Berkeley.

C'est une couche générale pour plusieurs bases de données sur fichiers. En tant que tel, les fonctionnalités sont limitées à une partie des fonctionnalités des bases de données modernes, comme Sleepycat Software's DB2. (A ne pas confondre avec IBM's DB2 software, qui fonctionne avec [10.69 ODBC.](#)).

Le comportement de certaines fonctions dépend de la base de données utilisée. Par exemple [dba_optimize\(\)](#) et [dba_sync\(\)](#) n'auront pas le même effet d'une base à l'autre.

Les bases suivantes sont supportées :

- dbm est la plus ancienne des base de données de type Berkeley. Il vaut mieux l'éviter si possible. Les fonctions de compatibilités codées dans DB2 et gdbm ne sont pas supportées, car elles ne sont compatibles qu'au niveau du code source, et ne peuvent pas gérer le format dbm originel.
- ndbm est un nouveau type de dbm plus flexible. Il a cependant la majorité des limitations du genre.
- gdbm est la base dbm GNU.
- db2 est DB2 de Sleepycat Software. Elle se décrit comme un "ensemble d'outils qui fournissent une base de données performante, tant pour les applications indépendantes que pour le client/serveur".
- cdb rdy "un package rapide, robuste, léger, pour créer et lire des bases de données constantes". C'est l'auteur de qmail qui l'a écrit, et elle est disponible [ici](#). Puisque c'est une base constante, elle ne supporte que la lecture.

Exemple DBA

```

<?php
$id = dba_open("/tmp/test.db", "n", "db2");
if(!$id) {
echo "dba_open a échoué\n";
exit;
}
dba_replace("key", "Ceci est un exemple!", $id);
if(dba_exists("key", $id)) {
echo dba_fetch("key", $id);
dba_delete("key", $id);
}
dba_close($id);
?>

```

DBA gère les données binaires, et n'a pas de limites arbitraires. Elle hérite de toutes les limites de la base sous jacentes.

Toutes les bases de données sur fichiers doivent fournir un moyen de changer le mode d'accès au fichier d'une base, et si possible, de toutes les bases. Le mode d'accès est généralement passé en 4ème argument à [dba_open\(\)](#) ou [dba_popen\(\)](#).

Vous pouvez accéder à toutes les entrées d'une base d'une manière linéaire, avec les fonctions [dba_firstkey\(\)](#) et [dba_nextkey\(\)](#). Vous ne devez pas modifier une base lorsque vous la traversez ainsi.

Traverser une base

```

<?php
# ...ouverture de la base...
$key = dba_firstkey($id);
while($key != false) {
if(...) { # conserver la clé pour faire d'autres opérations plus tard
    $handle_later[] = $key;
}
    $key = dba_nextkey($id);
}
for($i = 0; $i < count($handle_later); $i++)
dba_delete($handle_later[$i], $id);
?>

```

10.13.1 dba_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [dba_close](#)(int *handle*)

[dba_close\(\)](#) ferme le lien établi avec la base et libère toutes les ressources de *handle*.

handle est un identifiant de base, retourné par [dba_open\(\)](#).

[dba_close\(\)](#) ne retourne aucune valeur.

Voir aussi: [dba_open\(\)](#) et [dba_popen\(\)](#).

10.13.2 dba_delete

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dba_delete](#)(string *key*, int *handle*)

[dba_delete\(\)](#) efface l'entrée spécifiée par la clé *key*, dans la base identifiée par *handle*. *key* est la clé de l'entrée à effacer.

handle est un identifiant de lien, retourné par [dba_open\(\)](#).

[dba_delete\(\)](#) retourne TRUE ou FALSE, si l'entrée est effacée, ou pas effacée, respectivement.

Voir aussi: [dba_exists\(\)](#), [dba_fetch\(\)](#), [dba_insert\(\)](#) et [dba_replace\(\)](#).

[10.13.3 dba_exists](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dba_exists](#)(string *key*, int *handle*)

[dba_exists\(\)](#) vérifie si la clé *key* existe dans la base identifiée par *handle*.

key est la clé qui doit être vérifiée.

handle est un identifiant de base, retourné par [dba_open\(\)](#).

[dba_exists\(\)](#) retourne TRUE ou FALSE, si la clé est trouvée, ou pas, respectivement.

Voir aussi : [dba_fetch\(\)](#), [dba_delete\(\)](#), [dba_insert\(\)](#) et [dba_replace\(\)](#).

[10.13.4 dba_fetch](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dba_fetch](#)(string *key*, int *handle*)

[dba_fetch\(\)](#) lit les données spécifiée par la clé *key* dans la base identifiée par *handle*.

key est la clé dont on veut lire les données.

handle est un identifiant de base, retourné par [dba_open\(\)](#).

[dba_fetch\(\)](#) retourne la chaîne associée ou false, si la paire clé/valeur n'a pas été trouvée.

Voir aussi : [dba_exists\(\)](#), [dba_delete\(\)](#), [dba_insert\(\)](#) et [dba_replace\(\)](#).

[10.13.5 dba_firstkey](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dba_firstkey](#)(int *handle*)

[dba_firstkey\(\)](#) retourne la première clé de la base de données spécifiée par *handle* et y place le pointeur interne de clé. Cela permettra de traverser la base.

handle est un identifiant de base, retourné par [dba_open\(\)](#).

[dba_firstkey\(\)](#) retourne la clé, ou FALSE, suivant que la première clé existe ou pas.

Voir aussi : [dba_nextkey\(\)](#).

[10.13.6 dba_insert](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dba_insert](#)(string *key*, string *value*, int *handle*)

[dba_insert\(\)](#) insère l'entrée décrite par la clé *key* et la valeur *value* dans la base spécifiée par *handle*. Si une

entrée avec la même clé **key** existe déjà, l'insertion échouera.

key est la clé de la valeur à insérer.

value est la valeur à insérer.

handle est un identifiant de base, retourné par [dba_open\(\)](#).

[dba_insert\(\)](#) retourne true ou false, suivant que l'insertion a réussi ou échoué.

Voir aussi : [dba_exists\(\)](#), [dba_delete\(\)](#), [dba_fetch\(\)](#) et [dba_replace\(\)](#).

10.13.7 dba_nextkey

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dba_nextkey](#) (int **handle**)

[dba_nextkey\(\)](#) retourne la clé suivante, dans la base identifiée par **handle** et incrémente le pointeur de clé.

handle est un identifiant de base, retourné par [dba_open\(\)](#).

[dba_nextkey\(\)](#) retourne la clé, ou FALSE en cas d'échec.

Voir aussi: [dba_firstkey\(\)](#).

10.13.8 dba_popen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dba_popen](#) (string **path**, string **mode**, string **handler**, ...)

[dba_popen\(\)](#) établit une connexion persistante à la base repérée par **path** avec le mode **mode**, en utilisant l'identifiant **handler**.

path est le chemin sur votre machine.

mode vaut "r" pour lecture seule, "w" pour lecture/écriture, "c" pour lecture/écriture, et création si la base n'existe pas, et "n" pour création, écrasement, et accès en lecture/écriture.

handler est le nom de l'identifiant qui sera utilisé pour accéder à **path**. Il est passé à [dba_popen\(\)](#).

[dba_popen\(\)](#) retourne un identifiant positif, ou FALSE, suivant que la base a été ouverte, ou que l'accès a échoué.

Voir aussi : [dba_open\(\)](#) et [dba_close\(\)](#).

10.13.9 dba_open

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dba_open](#) (string **path**, string **mode**, string **handler**, ...)

[dba_open\(\)](#) établit une connexion à la base repérée par **path** avec le mode **mode** et l'identifiant **handler**.

path est le chemin sur votre machine.

mode vaut "r" pour lecture seule, "w" pour lecture/écriture, "c" pour lecture/écriture, et création si la base n'existe pas, et "n" pour création, écrasement, et accès en lecture/écriture.

handler est le nom de l'identifiant qui sera utilisé pour accéder à **path**. Il est passé à [dba_popen\(\)](#).

Voir aussi : [dba_popen\(\)](#) et [dba_close\(\)](#).

10.13.10 dba_optimize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dba_optimize](#) (int **handle**)

[dba_optimize\(\)](#) optimise la base de données identifiée par *handle*.

handle est un identifiant de base retourné par [dba_open\(\)](#).

[dba_optimize\(\)](#) retourne TRUE ou FALSE, suivant que l'optimisation a réussi ou échoué.

Voir aussi : [dba_sync\(\)](#).

[10.13.11 dba_replace](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dba_replace](#)(string *key*, string *value*, int *handle*)

[dba_replace\(\)](#) remplace ou insère une entrée, pour la clé *key* et avec la valeur *value* dans la base identifiée par *handle*.

key est la clé qui va être insérée.

value est la valeur qui va être insérée.

handle est un identifiant de base retourné par [dba_open\(\)](#).

[dba_replace\(\)](#) retourne true ou false, suivant que l'opération réussit ou échoue.

Voir aussi : [dba_exists\(\)](#), [dba_delete\(\)](#), [dba_fetch\(\)](#) et [dba_insert\(\)](#).

[10.13.12 dba_sync](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dba_sync](#)(int *handle*)

[dba_sync\(\)](#) synchronise la base de données spécifiée par *handle*. Si accepté, cela va probablement lancer une opération de réécriture physique du fichier.

handle est un identifiant de base retourné par [dba_open\(\)](#).

[dba_sync\(\)](#) retourne true ou false, si la synchronisation réussit, ou échoue, respectivement.

Voir aussi : [dba_optimize\(\)](#).

[10.14 Fonctions dBase](#)

[\[Notes en ligne\]](#)

Ces fonctions vous permettront d'accéder aux enregistrements d'une base au format dBase (.dbf).

dBase ne permet pas l'utilisation d'index, de "memo fields", ni le blocage de la base. Deux processus de serveurs web différents modifiant la même fichier dBase risque de rendre votre base de données incohérente.

A la différence des bases de données SQL, la définition des bases de données dBase, ne peut pas être changée. Une fois le fichier créé, la définition de la base est définitive. Il n'y a pas d'index qui accélèrent les recherches ou organisent vos données. Les fichiers dBase sont de simples fichiers séquentiels avec des enregistrements de longueur fixe. Les enregistrements sont ajoutés à la fin du fichier et les enregistrements supprimés sont conservés jusqu'à l'appel de [dbase_pack\(\)](#).

Nous vous recommandons de ne pas utiliser les fichiers dBase comme base de données de production.

Choisissez n'importe quel serveur SQL à la place. MySQL and Postgres sont des choix classiques avec PHP.

Le support de dBase ne se justifie ici que pour vous permettre d'importer et d'exporter des données de et vers votre base des données web, maintenant que le format du fichier est communément géré par les feuilles et organisateurs Windows. L'import et l'export de données est l'unique chose pour laquelle l'utilisation de dBase est recommandée.

10.14.1 dbase_create

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int dbase_create(string *filename*, array *fields*)

fields est un tableau de tableaux. Chaque tableau décrit le format d'un fichier de la base. Chaque champs est constitué d'un nom, d'un caractère de type de champs, d'une longueur et d'une précision.

Les types de champs disponibles sont :

L

- Boolean (booléen). Pas de longueur ou de précision pour ces valeurs.

M

- Memo. (Note importante : les Memos ne sont pas supportés par PHP.) Elles n'ont pas de longueur ou de précision.

D

- Date (enregistrée au format 'YYYYMMDD'). Elles n'ont pas de longueur ou de précision.

N

- Number (nombre). Possède une longueur et un précision (le nombre de chiffres après la virgule).

C

- String (chaîne).

Si la base de données a été créée, un identifiant de base `dbase_identifier` est retourné, sinon, `FALSE` est retourné. *Création d'une base dBase*

```
// "database" name
$dbname = "/tmp/test.dbf";
// database "definition"
$def =
array(
array("date",      "D"),
array("name",      "C",  50),
array("age",       "N",   3, 0),
array("email",     "C", 128),
array("ismember", "L")
);
// création
if (!dbase_create($dbname, $def))
print "<strong>Error!</strong>";
```

10.14.2 dbase_open

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int dbase_open(string *filename*, int *flags*)

flags est un flag, comme pour la fonction `open()`. (Typiquement; 0 signifie lecture seule, 1 signifie écriture seule, et 2 écriture/lecture).

Retourne un identifiant de base de données, ou FALSE si la base n'a pas pu être sélectionnée.

[10.14.3 dbase_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbase_close](#)(int *dbase_identifieur*)

Ferme la base associée à *dbase_identifieur*.

[10.14.4 dbase_pack](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbase_pack](#)(int *dbase_identifieur*)

Compacte la base de données spécifiée (effacement définitif de tous les enregistrements marqués pour l'effacement, avec la fonction [dbase_delete_record\(\)](#)).

[10.14.5 dbase_add_record](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbase_add_record](#)(int *dbase_identifieur*, array *record*)

Ajoute les données de *record* dans la base spécifiée par *dbase_identifieur*. Si le nombre de colonnes fourni n'est pas celui du nombre de champs dans la base, l'opération échouera, et FALSE sera retourné.

[10.14.6 dbase_replace_record](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbase_replace_record](#)(int *dbase_identifieur*, array *record*, int *dbase_record_number*)

Remplace les données associées à l'enregistrement *record_number* par les données enregistrées dans *record*. Si le nombre de colonnes fourni n'est pas celui du nombre de champs dans la base, l'opération échouera, et FALSE sera retourné.

dbase_record_number est un entier qui peut aller de 1 jusqu'au nombre maximal d'enregistrement de la base (retourné par [dbase_numrecords\(\)](#)).

[10.14.7 dbase_delete_record](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbase_delete_record](#)(int *dbase_identifieur*, int *record*)

Marque l'enregistrement *record* pour l'effacement, dans la base. Pour effacer réellement l'enregistrement, il faut utiliser aussi [dbase_pack\(\)](#).

[10.14.8 dbase_get_record](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [dbase_get_record](#)(int *dbase_identifieur*, int *record*)

Retourne les données de l'enregistrement *record* dans un tableau. Le tableau est indexé à partir de 0, et inclus un membre nommé 'deleted' (effacé), qui sera mis à 1 si l'enregistrement a été marqué pour l'effacement (voir [dbase_delete_record\(\)](#)).

Chaque champs est converti au format approprié PHP. (Les dates sont laissées au format chaîne).

[10.14.9 dbase_get_record_with_names](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [dbase_get_record_with_names](#)(int *dbase_identifieur*, int *record*)

Retourne les données de l'enregistrement *record* dans un tableau associatif. Le tableau inclus un membre nommé 'deleted' (effacé), qui sera mis à 1 si l'enregistrement a été marqué pour l'effacement (voir [dbase_delete_record\(\)](#)).

Chaque champs est converti au format approprié PHP. (Les dates sont laissées au format chaîne).

[10.14.10 dbase_numfields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dbase_numfields](#)(int *dbase_identifieur*)

Retourne le nombre de champs (colonnes) de la base de données spécifiée. Les numéros de champs sont numérotés de 0 à `dbase_numfields($db)-1`, tandis que les numéros d'enregistrements sont numérotés de 1 à `dbase_numrecords($db)`. *Utiliser [dbase_numfields\(\)](#)*

```
$rec = dbase_get_record($db, $recno);
$nf = dbase_numfields($db);
for ($i=0; $i < $nf; $i++) {
print $rec[$i]."<br>\n";
}
```

[10.14.11 dbase_numrecords](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dbase_numrecords](#)(int *dbase_identifieur*)

Retourne le nombre d'enregistrements (lignes) dans la base spécifiée. Les numéros de champs sont numérotés de 0 à `dbase_numfields($db)-1`, tandis que les numéros d'enregistrements sont numérotés de 1 à `dbase_numrecords($db)`.

10.15 Fonctions dbm

[\[Notes en ligne\]](#)

Ces fonctions vous permettent d'écrire des lignes dans une base de donnée de type dbm. Ce type de base (supporté par Berkeley db, gdbm, et quelques bibliothèques systèmes, ou certaines bibliothèques du système d'exploitation) enregistre les paires clés/valeurs, (contrairement aux enregistrements par ligne, utilisé par les autres bases de données relationnelles).

dbm

```
$dbm = dbmopen("dernier", "w");
if (dbmexists($dbm, $userid)) {
    $last_seen = dbmfetch($dbm, $userid);
} else {
    dbminsert($dbm, $userid, time());
}
faire_quelquechose();
dbmreplace($dbm, $userid, time());
dbmclose($dbm);
```

10.15.1 dbmopen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dbmopen](#)(string *filename*, string *flags*)

Le premier argument est le chemin absolu jusqu'au fichier dbm à ouvrir. Le deuxième argument est le mode d'ouverture du fichier, qui peut prendre les valeurs suivantes : "r", "n", "c" ou "w" qui représentent respectivement lecture seule, nouveau (ce qui implique lecture/écriture, et qui, probablement, va écraser une base existante), création(ce qui implique lecture/écriture, et qui, probablement, va écraser une base existante), et lecture/écriture.

Retourne un identifiant, qui sera passé à toutes les autres fonctions dbm, en cas de succès, ou FALSE en cas d'échec.

Si ndbm est utilisé, ndbm va créer les fichiers filename.dir et filename.pag. gdbm n'utilise qu'un fichier, tout comme les bibliothèques internes, et Berkeley db crée le fichier filename.db. Notez que PHP dispose de son propre système de verrouillage des fichiers, qui s'additionne à celui éventuellement fait par les bibliothèques. PHP n'efface jamais les fichiers .lck qu'il crée. Il les utilise comme inode fixe, sur lequel faire le verrouillage . Pour plus d'informations sur les fichiers dbm, reportez vous à vos pages de manuel Unix (man) , ou bien chargez gdbm : `ftp://prep.ai.mit.edu/pub/gnu'.

10.15.2 dbmclose

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbmclose](#)(int *dbm_identifiant*)

Déverrouille et ferme la base de données dbm_identifiant.

10.15.3 dbmexists

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbmexists](#) (int *dbm_identifieur*, string *key*)

Retourne TRUE si il y a une valeur associée à la clé *key*.

10.15.4 dbmfetch

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dbmfetch](#) (int *dbm_identifieur*, string *key*)

Retourne la valeur associée à la clé *key*.

10.15.5 dbminsert

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dbminsert](#) (int *dbm_identifieur*, string *key*, string *value*)

Ajoute la valeur *value* dans la base de données, avec la clé *key*.

Retourne -1 si la base a été ouverte en mode lecture seule ;, 0 si l'insertion a été réussie, et 1 si la clé *key* existe déjà. (Pour remplacer la valeur, utilisez [dbmreplace\(\)](#).)

10.15.6 dbmreplace

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbmreplace](#) (int *dbm_identifieur*, string *key*, string *value*)

Remplace une valeur courante par la valeur *value* pour la clé *key*, dans une base dbm. Cette fonction va créer la clé, si elle n'existe pas dans la base.

10.15.7 dbmdelete

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbmdelete](#) (int *dbm_identifieur*, string *key*)

Efface la valeur de la clé *key*, dans la base dbm.

Retourne FALSE si la clé n'existe pas dans cette base.

10.15.8 dbmfirstkey

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dbmfirstkey](#) (int *dbm_identifieur*)

Retourne la première clé de la base de données. Notez bien que les clés ne sont pas dans un ordre défini, étant donné que la table est construite comme une table de hash.

10.15.9 dbmnextkey

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dbmnextkey](#)(int *dbm_identifieur*, string *key*)

Retourne la clé après la clé *key*. En appelant [dbmfirstkey\(\)](#), puis successivement [dbmnextkey\(\)](#), il est possible de passer en revue toute les paires clé/valeur de la base de données dbm. Par exemple : *Passer en revue une base de données.*

```
$key = dbmfirstkey($dbm_id);
while ($key) {
echo "$key = " . dbmfetch($dbm_id, $key) . "\n";
    $key = dbmnextkey($dbm_id, $key);
}
```

10.15.10 dblist

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dblist](#)(void)

10.16 Accès aux dossiers

[\[Notes en ligne\]](#)

10.16.1 chdir

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [chdir](#)(string *directory*)

Change le dossier courant de PHP en *directory*. Retourne FALSE si l'opération échoue, et TRUE sinon.

10.16.2 dir

[\[Notes en ligne\]](#) [\[Exemples\]](#)

new [dir](#)(string *directory*)

Un mécanisme pseudo-objet permet la lecture d'un dossier. L'argument *directory* doit être ouvert. Deux propriétés sont disponibles une fois le dossier ouvert : le pointeur peut être utilisé avec d'autres fonctions telles que [readdir\(\)](#), [rewinddir\(\)](#) et [closedir\(\)](#). Le chemin du dossier est le chemin fourni lors de la construction de l'objet. Trois méthodes permettent de lire, remettre à zéro et fermer le dossier. [dir\(\)](#) *example*

```
$d = dir("/etc");
echo "Pointeur: " . $d->handle . "<br>\n";
```

```

echo "Chemin: ".$d->path."<br>\n";
while($entry=$d->read()) {
echo $entry."<br>\n";
}
$d->close();

```

[10.16.3 closedir](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [closedir](#)(int *dir_handle*)

[closedir\(\)](#) ferme le pointeur de dossier *dir_handle*. Le dossier devait avoir été ouvert avec [opendir\(\)](#).

[10.16.4 opendir](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [opendir](#)(string *path*)

[opendir\(\)](#) retourne un pointeur sur un dossier pour être utilisé avec les fonctions [closedir\(\)](#), [readdir\(\)](#) et [rewinddir\(\)](#).

[10.16.5 readdir](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [readdir](#)(int *dir_handle*)

[readdir\(\)](#) retourne le nom du fichier suivant dans le dossier identifié par *dir_handle*. Les noms sont retournés dans n'importe quel ordre. *Liste tous les fichiers du dossier courant*

```

<?php
$handle=opendir('.');
echo "Pointeur de dossier: $handle\n";
echo "Fichiers:\n";
while ($file = readdir($handle)) {
echo "$file\n";
}
closedir($handle);
?>

```

Notez que [readdir\(\)](#) retournera aussi les dossiers "." et "..". Si vous ne les voulez pas, supprimez les simplement : *Liste tous les fichiers du dossier courant, sauf . et ..* **Liste tous les fichiers du dossier courant, sauf . et ..**

```

<?php
$handle=opendir('.');

```

```

while ($file = readdir($handle)) {
if ($file != "." 38;$file != "..") {
echo "$file\n";
}
}
closedir($handle);
?>

```

10.16.6 rewinddir

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [rewinddir](#) (int *dir_handle*)

[rewinddir\(\)](#) retourne à la première entrée du dossier : le prochain fichier lu sera le premier.

10.17 Chargement dynamique de fonctions

[\[Notes en ligne\]](#)

10.17.1 dl

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dl](#) (string *library*)

Charge l'extension PHP définie dans la bibliothèque *library*. Voir aussi la directive de configuration [7.1.5.2 ini.extension_dir](#).

10.18 Fonctions DOM XML

[\[Notes en ligne\]](#)

Ces fonctions ne sont disponibles que si PHP a été configuré avec l'option `--with-dom=[DIR]`, et utilise la librairie GNOME xml library. Vous aurez aussi besoin de la librairie libxml-2.0.0 (la version beta ne fonctionne pas). Ces fonctions ont été ajoutées dans PHP4.

Note du traducteur : liste des fonctions à venir (13/6/2000) domxml_root domxml_add_root domxml_dumpmem domxml_attributes domxml_getattr domxml_setattr domxml_children domxml_new_child domxml_node domxml_new_xmldoc alias new_xmldoc.

Ce module définit les constantes suivantes :

Constant	Valeur	Description
XML_ELEMENT_NODE	1	@tab
XML_ATTRIBUTE_NODE	2	@tab
XML_TEXT_NODE	3	@tab
XML_CDATA_SECTION_NODE	4	@tab
XML_ENTITY_REF_NODE	5	@tab

XML_ENTITY_NODE	6	@tab
XML_PI_NODE	7	@tab
XML_COMMENT_NODE	8	@tab
XML_DOCUMENT_NODE	9	@tab
XML_DOCUMENT_TYPE_NODE	10	@tab
XML_DOCUMENT_FRAG_NODE	11	@tab
XML_NOTATION_NODE	12	@tab
XML_GLOBAL_NAMESPACE	1	@tab
XML_LOCAL_NAMESPACE	2	@tab

Ce module définit un ensemble de classes. Les fonctions DOM XML retourne un arbre à partir d'un document XML, dont chaque noeud est un objet issu de ces classes.

[10.18.1 xmldoc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [xmldoc](#) (string *str*)

[xmldoc\(\)](#) analyse le document XML *str* et retourne un objet de classe "Dom document", avec les propriétés de "doc" (ressources), "version" (string) et "type" (long).

[10.18.2 xmldocfile](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [xmldocfile](#) (string *filename*)

[xmldocfile\(\)](#) analyse le fichier XML *filename* et retourne un objet "Dom document", avec les propriétés de "doc" (ressources) et "version" (string).

[10.18.3 xmltree](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [xmltree](#) (string *str*)

[xmltree\(\)](#) analyse le document XML *str* et retourne un arbre d'objets PHP qui représente le document analysé.

[10.19 Error Handling and Logging Functions](#)

[\[Notes en ligne\]](#)

These are functions dealing with error handling and logging. They allow you to define your own error handling rules, as well as modify the way the errors can be logged. This allows you to change and enhance error reporting to suit your needs.

With the logging functions, you can send messages directly to other machines, to an email (or email to pager

gateway!), to system logs, etc., so you can selectively log and monitor the most important parts of your applications and websites.

The error reporting functions allow you to customize what level and kind of error feedback is given, ranging from simple notices to customized functions returned during errors.

[10.19.1 error_log](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [error_log](#) (string *message*, int *message_type*, string *destination* , string *extra_headers*)

Envoie un message d'erreur dans les logs du serveur, sur un port *TCP*, ou bien à un fichier. Le premier paramètre, *message*, est le message qui doit être envoyé. Le deuxième paramètre, *message_type* indique l'endroit où le message doit aller:

0	Le message <i>message</i> est envoyé au système de log du PHP, en utilisant la fonction système de logs, en fonction de la configuration de l'option 7.1.1.8 ini.error_log . @tab
1	<i>message</i> Le message est envoyé par mail à l'adresse donnée dans le paramètre <i>destination</i> . C'est le seul moment où le quatrième paramètres, <i>extra_headers</i> est utilisé. Ce type de message utilise les mêmes fonctions internes que la fonction mail() . @tab
2	Le message <i>message</i> est envoyé au travers de la connexion de débogage. Cette option est disponible uniquement si <code>@xref{enable-debugger}</code> a été activée. Dans ce cas, le paramètre <i>destination</i> spécifie le nom de l'hôte ou l'adresse IP ainsi que, optionnellement, le numéro de port de la socket recevant les informations de débogage. @tab
3	Le message <i>message</i> est simplement écrit dans le fichier <i>destination</i> . @tab

Exemple avec [error_log\(\)](#)

```
// Envoie un message à l'administrateur si on ne peut
// pas se connecter à la base de donnée.
if (!Ora_Logon($username, $password)) {
error_log("Base de données Oracle indisponible!", 0);
}
// Envoie un mail à l'administrateur.
if (!$foo = allocate_new_foo()) {
error_log("Gros problème de connexion à la base!", 1,
"operator@mydomain.com");
}
// Une autre manière d'appeler la fonction error_log():
error_log("Grosse bourde!", 2, "127.0.0.1:7000");
error_log("Grosse bourde!", 2, "loghost");
error_log("Grosse bourde!", 3, "/var/tmp/my-errors.log");
```

[10.19.2 error_reporting](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [error_reporting](#) (int *level*)

établit le niveau d'erreur à prendre en compte et renvoie l'ancien niveau. Le niveau d'erreur est un champs de bits qui utilise les valeurs suivantes: (suivez les liens pour connaître les valeurs et leur signification)

Valeur	Nom interne
1	C.3.3 E_ERROR @tab
2	C.3.2 E_WARNING @tab
4	C.3.4 E_PARSE @tab
8	C.3.1 E_NOTICE @tab
16	E_CORE_ERROR @tab
32	E_CORE_WARNING @tab

[10.19.3 restore_error_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [restore_error_handler](#) (void)

Used after changing the error handler function using [set_error_handler\(\)](#), to revert to the previous error handler (which could be the built-in or a user defined function)

See also [error_reporting\(\)](#), [set_error_handler\(\)](#), [trigger_error\(\)](#), [user_error\(\)](#)

[10.19.4 set_error_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [set_error_handler](#) (string *error_handler*)

Sets a user function (*error_handler*) to handle errors in a script. Returns the previously defined error handler (if any), or false on error. This function can be used for defining your own way of handling errors during runtime, for example in applications in which you need to do cleanup of data/files when a critical error happens, or when you need to trigger an error under certain conditions (using [trigger_error\(\)](#))

The user function needs to accept 2 parameters: the error code, and a string describing the error. The example below shows the handling of internal exceptions by triggering errors and handling them with a user defined function: error handling with [set_error_handler\(\)](#) and [trigger_error\(\)](#) *error handling with [set_error_handler\(\)](#) and [trigger_error\(\)](#)*

```
<?php
// redefine the user error constants - PHP 4 only
define (FATAL,E_USER_ERROR);
define (ERROR,E_USER_WARNING);
define (WARNING,E_USER_NOTICE);
// set the error reporting level for this script
error_reporting (FATAL + ERROR + WARNING);
```

```

// error handler function
function myErrorHandler ($errno, $errstr) {
switch ($errno) {
case FATAL:
echo "<b>FATAL</b> [$errno] $errstr<br>\n";
echo " Fatal error in line ".__LINE__." of file ".__FILE__;
echo ", PHP ".PHP_VERSION." (" .PHP_OS.")<br>\n";
echo "Aborting...<br>\n";
exit -1;
break;
case ERROR:
echo "<b>ERROR</b> [$errno] $errstr<br>\n";
break;
case WARNING:
echo "<b>WARNING</b> [$errno] $errstr<br>\n";
break;
default:
echo "Unkown error type: [$errno] $errstr<br>\n";
break;
}
}
// function to test the error handling
function scale_by_log ($vect, $scale) {
if ( !is_numeric($scale) || $scale <= 0 )
trigger_error("log(x) for x <= 0 is undefined, you used: scale = $scale",
FATAL);
if (!is_array($vect)) {
trigger_error("Incorrect input vector, array of values expected", ERROR);
return null;
}
for ($i=0; $i<count($vect); $i++) {
if (!is_numeric($vect[$i]))
trigger_error("Value at position $i is not a number, using 0 (zero)",
WARNING);
$temp[$i] = log($scale) * $vect[$i];
}
return $temp;
}
// set to the user defined error handler
$old_error_handler = set_error_handler("myErrorHandler");
// trigger some errors, first define a mixed array with a non-numeric item
echo "vector a\n";
$a = array(2,3,"foo",5.5,43.3,21.11);
print_r($a);
// now generate second array, generating a warning
echo "----\nvector b - a warning (b = log(PI) * a)\n";
$b = scale_by_log($a, M_PI);
print_r($b);
// this is trouble, we pass a string instead of an array
echo "----\nvector c - an error\n";
$c = scale_by_log("not array",2.3);
var_dump($c);
// this is a critical error, log of zero or negative number is undefined
echo "----\nvector d - fatal error\n";
$d = scale_by_log($a, -2.5);
?>

```

And when you run this sample script, the output will be

```

vector a
Array

```

```
(
  [0] => 2
  [1] => 3
  [2] => foo
  [3] => 5.5
  [4] => 43.3
  [5] => 21.11
)
-----
vector b - a warning (b = log(PI) * a)
<b>WARNING</b> [1024] Value at position 2 is not a number, using 0 (zero)<br>
Array
(
  [0] => 2.2894597716988
  [1] => 3.4341896575482
  [2] => 0
  [3] => 6.2960143721717
  [4] => 49.566804057279
  [5] => 24.165247890281
)
-----
vector c - an error
<b>ERROR</b> [512] Incorrect input vector, array of values expected<br>
NULL
-----
vector d - fatal error
<b>FATAL</b> [256] log(x) for x <= 0 is undefined, you used: scale = -2.5<br>
Fatal error in line 16 of file trigger_error.php, PHP 4.0.1pl2 (Linux)<br>
Aborting...<br>
```

See also [error_reporting\(\)](#), [restore_error_handler\(\)](#), [trigger_error\(\)](#), [user_error\(\)](#)

10.19.5 trigger_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [trigger_error](#)(string *error_msg*, int *error_type*)

Used to trigger a user error condition, it can be used by in conjunction with the built-in error handler, or with a user defined function that has been set as the new error handler ([set_error_handler\(\)](#)). This function is useful when you need to generate a particular response to an exception at runtime. For example:

```
if (assert ($divisor == 0))
trigger_error ("Cannot divide by zero", E_USER_ERROR);
```

Note : See [set_error_handler\(\)](#) for a more extensive example.

See also [error_reporting\(\)](#), [set_error_handler\(\)](#), [restore_error_handler\(\)](#), [user_error\(\)](#)

10.19.6 user_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [user_error](#)(string *error_msg*, int *error_type*)

This is an alias for the function [trigger_error\(\)](#).

See also [error_reporting\(\)](#), [set_error_handler\(\)](#), [restore_error_handler\(\)](#), and [trigger_error\(\)](#).

10.20 Fonction d'exécution de programmes

[\[Notes en ligne\]](#)

10.20.1 escapeshellcmd

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [escapeshellcmd](#) (string *command*)

[escapeshellcmd\(\)](#) échappe tous les caractères de la chaîne *command* qui pourraient avoir une signification spéciale dans une commande shell. Cette fonction permet de s'assurer que la commande sera correctement passée à l'exécuteur de commande shell [exec\(\)](#) et [system\(\)](#), ou encore à [9.7.6 Opérateur d'exécutions](#). Généralement, cette fonction est utilisée comme ceci :

```
system( EscapeShellCmd( $cmd ) )
```

Voir aussi [exec\(\)](#), [popen\(\)](#), [system\(\)](#), et les [9.7.6 Opérateur d'exécutions](#).

10.20.2 exec

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [exec](#) (string *command*, string *array* , int *return_var*)

[exec\(\)](#) exécute la commande *command*, mais ne renvoie rien comme retour, hormis la dernière ligne du résultat de la commande. Pour exécuter une commande et obtenir le résultat sans aucun traitement, il faut utiliser la fonction [passthru\(\)](#).

Si l'argument *array* est présent, alors ce tableau sera rempli par les lignes retournées par la commande. Il faut noter que si ce tableau contient des éléments, [exec\(\)](#) ajoutera les nouvelles lignes à la fin du tableau. Si vous ne voulez pas que les nouveaux éléments soient concaténés, utilisez la fonction [unset\(\)](#) avec ce tableau avant de le passer à [exec\(\)](#).

Si l'argument *return_var* est présent en plus du tableau *array*, alors de statut de retour d'exécution sera inscrit dans cette variable.

Notez que si vous allez fournir des commandes qui proviennent d'un utilisateur, il est avisé d'utiliser la fonction [escapeshellcmd\(\)](#) pour s'assurer que l'utilisateur n'essaie pas de profiter des caractères spéciaux pour tromper le système.

Voir aussi [system\(\)](#), [passthru\(\)](#), [popen\(\)](#), [escapeshellcmd\(\)](#), et les [9.7.6 Opérateur d'exécutions](#).

10.20.3 passthru

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [passthru](#) (string *command*, int *return_var*)

La fonction [passthru\(\)](#) est similaire à la fonction [exec\(\)](#) car les deux exécutent la commande *command*. Si l'argument *return_var* est présent, le code de statut de réponse UNIX y sera placé. Cette fonction doit être utilisée de préférence aux commandes [exec\(\)](#) ou [system\(\)](#) lorsque le résultat attendu est de type binaire, et doit être passé tel quel à un navigateur. Une utilisation classique de cette fonction est l'exécution de l'utilitaire

pbmplus qui peut retourner une image. En fixant le résultat du contenu (content-type) à "image/gif" puis en appelant pbmplus pour obtenir une image gif, vous pouvez créer des scripts PHP qui retourne des images. Voir aussi [exec\(\)](#), [system\(\)](#), [popen\(\)](#), [escapeshellcmd\(\)](#), et les [9.7.6 Opérateur d'exécutions](#).

10.20.4 system

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [system](#) (string *command*, int *return_var*)

[system\(\)](#) est la version PHP de la fonction C qui exécute la commande *command* et retourne le résultat. Si une variable est fournie comme second argument, alors le code de statut de la commande y sera affecté. Notez que si vous allez fournir des commandes qui proviennent d'un utilisateur, il est avisé d'utiliser la fonction [escapeshellcmd\(\)](#) pour s'assurer que l'utilisateur n'essaie pas de profiter des caractères spéciaux pour tromper le système.

[system\(\)](#) essaie automatiquement de vider les tampons du serveur web après chaque ligne de résultat PHP, lorsque ce dernier fonctionne comme un module.

Retourne la dernière ligne du retour, en cas de succès, et FALSE en cas d'échec.

Si vous devez exécuter une commande et récupérer tout le résultat dans aucune intervention, utilisez la fonction [passthru\(\)](#).

Voir aussi [exec\(\)](#), [passthru\(\)](#), [popen\(\)](#), [escapeshellcmd\(\)](#) et les [9.7.6 Opérateur d'exécutions](#).

10.21 Forms Data Format

[\[Notes en ligne\]](#)

Forms Data Format (FDF) est un format de formulaire pour les documents PDF. Vous pouvez lire la documentation (en anglais) à <http://partners.adobe.com/asn/developer/acrosdk/forms.html> pour plus de détails sur les tenants et les aboutissants.

Note : Actuellement, Adobe ne fournit que *libc5*, compatible Linux. Les tests avec *glibc2* ont donné des "segmentation fault". Si quelqu'un est arrivé à un résultat, envoyez nous vos infos.

Note : Si vous rencontrez des problèmes de configuration avec *fdftk*, vérifiez que le header *FdfTk.h* et la librairie *libFdfTk.so* sont bien là où ils doivent être. Ils devraient être placés dans le dossier ``fdftk-dir/include'` et ``fdftk-dir/lib'`. Cela ne sera pas le cas si vous avez simplement décompressé la distribution.

L'esprit de FDF est similaire à celui des formulaires HTML. Les différences résident dans les moyens de transmission des données au serveur, lorsque le bouton "submit" (soumettre) est pressé (ce qui est du ressort de Form Data Format) et le format de formulaire lui-même (qui est plutôt du ressort de Portable Document Format, PDF). Gérer des données FDF est un des objectifs des fonctions FDF. Mais il y en a d'autres. Vous pouvez aussi prendre un formulaire PDF, et pré-remplir les champs, sans modifier le formulaire lui-même. Dans ce cas, on va créer un document FDF ([fdf_create\(\)](#)), remplir les champs ([fdf_set_value\(\)](#)) et l'associer à un fichier PDF ([fdf_set_file\(\)](#)). Finalement, le tout sera envoyé au client, avec le type MIME "application/vnd.fdf". Le module "Acrobat reader" de votre navigateur va reconnaître ce type MIME, et lire le fichier PDF, puis le remplir avec FDF.

Les exemples suivants montrent comment évaluer les données du formulaire.

Evaluer un document FDF

```
<?php
```

```
// Sauver le fichier FDF dans un fichier temporaire.
$fdffp = fopen("test.fdf", "w");
fwrite($fdffp, $HTTP_FDF_DATA, strlen($HTTP_FDF_DATA));
fclose($fdffp);
// Ouvrir le fichier temporaire, et utiliser les données.
// Le formulaire pdf contenait différents fichiers texte, avec pour nom :
// volume, date, comment, publisher, preparer, ainsi que deux boîtes à cocher,
// show_publisher et show_preparer.
$fdf = fdf_open("test.fdf");
$volume = fdf_get_value($fdf, "volume");
echo "La valeur du champs volume était : '<B>$volume</B>'\<BR>";
$date = fdf_get_value($fdf, "date");
echo "La valeur du champs date était '<B>$date</B>'\<BR>";
$comment = fdf_get_value($fdf, "comment");
echo "La valeur du champs comment était '<B>$comment</B>'\<BR>";
if(fdf_get_value($fdf, "show_publisher") == "On") {
    $publisher = fdf_get_value($fdf, "publisher");
    echo "La valeur du champs publisher était : '<B>$publisher</B>'\<BR>";
} else
echo "La valeur du champs ne doit pas être affichée.<BR>";
if(fdf_get_value($fdf, "show_preparer") == "On") {
    $preparer = fdf_get_value($fdf, "preparer");
    echo "La valeur du champs preparer était '<B>$preparer</B>'\<BR>";
} else
echo "La valeur du champs Preparer ne doit pas être affiché.<BR>";
fdf_close($fdf);
?>
```

[10.21.1 fdf_open](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fdf_open](#)(string *filename*)

La fonction [fdf_open\(\)](#) ouvre un fichier avec formulaire. Le fichier doit contenir les données retournées par le formulaire PDF. Actuellement, le fichier doit être créée 'manuellement', en utilisant la fonction [fopen\(\)](#) et en y écrivant le contenu du tableau HTTP_FDF_DATA avec la fonction [fwrite\(\)](#). Un mécanisme comparable aux formulaires HTML qui créent une variable pour chaque champs entrant, n'existe pas.

Accéder aux données du formulaire

```
<?php
// Sauver le fichier FDF dans un fichier temporaire.
$fdffp = fopen("test.fdf", "w");
fwrite($fdffp, $HTTP_FDF_DATA, strlen($HTTP_FDF_DATA));
fclose($fdffp);
// Ouvrir le fichier temporaire, et utiliser les données.
$fdf = fdf_open("test.fdf");
...
fdf_close($fdf);
?>
```

Voir aussi [fdf_close\(\)](#).

[10.21.2 fdf_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [fdf_close](#) (int *fdf_document*)

[fdf_close\(\)](#) ferme le document FDF.

Voir aussi [fdf_open\(\)](#).

[10.21.3 fdf_create](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fdf_create](#) (void)

[fdf_create\(\)](#) crée un nouveau document FDF. Cette fonction est nécessaire pour ceux qui veulent pré remplir les champs d'un formulaire dans un fichier PDF.

Pré remplir un formulaire PDF

```

<?php
$outfdf = fdf_create();
fdf_set_value($outfdf, "volume", $volume, 0);
fdf_set_file($outfdf, "http://testfdf/resultlabel.pdf");
fdf_save($outfdf, "outtest.fdf");
fdf_close($outfdf);
Header("Content-type: application/vnd.fdf");
$fp = fopen("outtest.fdf", "r");
fpassthru($fp);
unlink("outtest.fdf");
?>

```

Voir aussi [fdf_close\(\)](#), [fdf_save\(\)](#), et [fdf_open\(\)](#).

[10.21.4 fdf_save](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fdf_save](#) (string *filename*)

[fdf_save\(\)](#) sauve un document FDF. Le FDF Toolkit fournit un moyen d'envoyer le contenu d'un document FDF à au fichier de sortie stdout si le paramètre *filename* vaut '.'. Ceci ne fonctionne pas si PHP est sous la forme d'un module Apache. Dans ce cas, il faudra écrire le résultat dans un fichier, et utiliser [fpassthru\(\)](#) pour l'afficher au client.

Voir aussi [fdf_close\(\)](#) et pour avoir un exemple [fdf_create\(\)](#).

[10.21.5 fdf_get_value](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fdf_get_value](#) (int *fdf_document*, string *fieldname*)

[fdf_get_value\(\)](#) retourne la valeur d'un champs.

Voir aussi [fdf_set_value\(\)](#).

10.21.6 fdf_set_value

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [fdf_set_value](#)(int *fdf_document*, string *fieldname*, string *value*, int *isName*)

[fdf_set_value\(\)](#) fixe la valeur d'un champs. Le dernier paramètre détermine si la valeur doit être convertie en nom PDF (*isName* = 1) ou affecter une chaîne PDF à un contrôle (*isName* = 0).

Voir aussi [fdf_get_value\(\)](#).

10.21.7 fdf_next_field_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fdf_next_field_name](#)(int *fdf_document*, string *fieldname*)

[fdf_next_field_name\(\)](#) retourne le nom du champs après le champs *fieldname* ou le nom du premier champs, si le second paramètre est NULL.

Voir aussi [fdf_set_value\(\)](#) et [fdf_get_value\(\)](#).

10.21.8 fdf_set_ap

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [fdf_set_ap](#)(int *fdf_document*, string *field_name*, int *face*, string *filename*, int *page_number*)

[fdf_set_ap\(\)](#) fixe l'apparence d'un champs (i.e. la valeur de la clé /AP). Les valeurs possibles de *face* sont 1=FDFNormalAP, 2=FDFRolloverAP, 3=FDFDownAP.

10.21.9 fdf_set_status

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [fdf_set_status](#)(int *fdf_document*, string *status*)

[fdf_set_status\(\)](#) fixe la valeur de la clé /STATUS.

Voir aussi [fdf_get_status\(\)](#).

10.21.10 fdf_get_status

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fdf_get_status](#)(int *fdf_document*)

[fdf_get_status\(\)](#) retourne la valeur de la clé /STATUS.

Voir aussi [fdf_set_status\(\)](#).

[10.21.11 fdf_set_file](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [fdf_set_file](#)(int *fdf_document*, string *filename*)

[fdf_set_file\(\)](#) Fixe la valeur de la clé /F. la clé /F est simplement une référence sur un formulaire PDF qui doit être pré-remplis. Dans un environnement web, c'est une URL (e.g. <http://testfdf/resultlabel.pdf>). Voir aussi [fdf_get_file\(\)](#) et pour un exemple, [fdf_create\(\)](#).

[10.21.12 fdf_get_file](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fdf_get_file](#)(int *fdf_document*)

[fdf_get_file\(\)](#) lit la valeur de la clé /F.
Voir aussi [fdf_set_file\(\)](#).

[10.22 Fonctions filePro](#)

[\[Notes en ligne\]](#)

Ces fonctions permettent de lire des données enregistrées dans des bases non modifiables, sur des serveurs filePro.

filePro est une marque de Fiserv, Inc. Vous pouvez avoir plus de détails sur filePro à <http://www.fileproplus.com/>.

[10.22.1 filepro](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [filepro](#)(string *directory*)

Cette fonction lit et vérifie un fichier, puis enregistre le nombre de champs et de lignes.

Aucun verrouillage n'est pratiqué : il vaut alors mieux ne pas modifier la base filePro lorsqu'elle est ouverte par PHP.

[10.22.2 filepro_fieldname](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [filepro_fieldname](#)(int *field_number*)

Retourne le nom du champs d'index *field_number*.

[10.22.3 filepro_fieldtype](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [filepro_fieldtype](#)(int *field_number*)

Retourne le type du champs d'index *field_number*.

[10.22.4 filepro_fieldwidth](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filepro_fieldwidth](#)(int *field_number*)

Retourne la taille du champs d'index *field_number*.

[10.22.5 filepro_retrieve](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [filepro_retrieve](#)(int *row_number*, int *field_number*)

Retourne la valeur du champs d'index *row_number*, et à la ligne *field_number*.

[10.22.6 filepro_fieldcount](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filepro_fieldcount](#)

Retourne le nombre de champs (ou colonnes) d'une base filePro.

Voir aussi [filepro\(\)](#).

[10.22.7 filepro_rowcount](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filepro_rowcount](#)

Retourne le nombre de lignes dans une base filePro.

Voir aussi [filepro\(\)](#).

[10.23 Système de fichiers](#)

[\[Notes en ligne\]](#)

[10.23.1 basename](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [basename](#)(string *path*)

Cette fonction prend en paramètre le chemin complet d'un fichier et en extrait le nom du fichier.

Sous Windows, les caractères (/) et backslash (\) sont utilisés comme séparateur de dossier. Sous les autres OS, seul le caractère slash (/) est utilisé.

Exemple avec [basename\(\)](#)

```
$path = "/home/httpd/html/index.php3";
$file = basename($path); // $file est affecté avec "index.php3"
```

Voir aussi: [dirname\(\)](#).

10.23.2 chgrp

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [chgrp](#)(string *filename*, mixed *group*)

[chgrp\(\)](#) essaie de changer le groupe propriétaire du fichier. Seul le superuser (root) peut changer le groupe propriétaire d'un fichier arbitrairement. Les utilisateurs classiques ne peuvent changer le groupe propriétaire d'un fichier que si l'utilisateur propriétaire du fichier est membre du groupe.

Renvoie TRUE en cas de succès, sinon renvoie FALSE.

Voir aussi [chown\(\)](#) et [chmod\(\)](#).

Note : *Sous Windows, cette fonction ne fait rien et renvoie TRUE dans tous les cas.*

10.23.3 chmod

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [chmod](#)(string *filename*, int *mode*)

[chmod\(\)](#) remplace le mode du fichier *filename* par le mode *mode*.

Il est à noter que le mode *mode* est considéré comme un nombre en notation octale. Afin de vous en assurer, vous pouvez préfixer cette valeur par un zéro (*mode*):

```
chmod( "/somedir/somefile", 755 ); // notation décimale; probablement FALSE
chmod( "/somedir/somefile", 0755 ); // notation octale; valeur du mode correcte
```

Renvoie TRUE en cas de succès, FALSE sinon.

Voir aussi [chown\(\)](#) et [chgrp\(\)](#).

Note : *Cette fonction ne fonctionne pas sous Windows.*

10.23.4 chown

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [chown](#)(string *filename*, mixed *user*)

[chown\(\)](#) change le groupe propriétaire du fichier. Seul le superutilisateur (root) peut changer le propriétaire d'un fichier.

Renvoie "TRUE" en cas de succès, "FALSE" sinon. Note : *Sous Windows, ne fait rien et retourne TRUE.*

Voir aussi [chown\(\)](#) et [chmod\(\)](#).

Note : *Cette fonction est inopérante sous Windows*

10.23.5 clearstatcache

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [clearstatcache](#)

L'appel à la fonction `stat` ou `lstat` est relativement couteux en terme de temps d'exécution. Pour cela, le résultat du dernier appel à l'une des fonctions de statut, (voir la liste ci-dessous), est sauvegardé pour ré-utilisation ultérieure. Si vous voulez forcer la vérification du statut d'un fichier, dans le cas où le fichier aurait pu être modifié ou aurait disparu, vous devez utiliser la fonction [clearstatcache\(\)](#) afin d'effacer de la mémoire les résultats du dernier appel à la fonction.

La valeur du cache n'est valable que pour la durée d'une requête.

Les fonctions affectées sont : [stat\(\)](#), [lstat\(\)](#), [file_exists\(\)](#), [is_writeable\(\)](#), [is_readable\(\)](#), [is_executable\(\)](#), [is_file\(\)](#), [is_dir\(\)](#), [is_link\(\)](#), [filectime\(\)](#), [fileatime\(\)](#), [filemtime\(\)](#), [fileinode\(\)](#), [filegroup\(\)](#), [fileowner\(\)](#), [filesize\(\)](#), [filetype\(\)](#), et [fileperms\(\)](#).

10.23.6 copy

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [copy](#)(string *source*, string *dest*)

[copy\(\)](#) fait une copie du fichier. Renvoie TRUE en cas de succès, FALSE sinon. *Exemple avec [copy\(\)](#)*

```
if ( !copy($file, $file.'.bak') ) {
print("La copie du fichier $file n'a pas réussi...<br>\n");
}
```

Voir aussi: [rename\(\)](#).

10.23.7 delete

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [delete](#)(string *file*)

Ceci est une fausse entrée du manuelle pour ceux qui recherchent en fait la fonction [unlink\(\)](#) ou [unset\(\)](#). Voir aussi: [unlink\(\)](#) pour effacer des fichiers, [unset\(\)](#) pour effacer des variables.

10.23.8 dirname

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dirname](#)(string *path*)

Si *path* contient le chemin d'un fichier ou dossier, cette fonction retournera le nom du dossier qui le contient. Sous Windows, les slash (/) et backslash (\) sont utilisés comme séparateurs de dossier. Dans les autres environnements, seul le slash (/) est utilisé.

Exemple avec [dirname\(\)](#)

```
$path = "/etc/passwd";
$file = dirname($path); // $file contient "/etc"
```

Voir aussi: [basename\(\)](#).

[10.23.9 diskfreespace](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [diskfreespace](#) (string *directory*)

[diskfreespace\(\)](#) retournera le nombre d'octets disponible sur le disque correspondant contenant le dossier *directory*.

Exemple avec [diskfreespace\(\)](#)

```
$df = diskfreespace("/"); // $df contient le nombre d'octets libres sur "/"
```

[10.23.10 fclose](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fclose](#) (int *fp*)

Ferme le fichier *fp*.

Retourne TRUE en cas de succès, et FALSE en cas d'échec.

Le pointeur de fichier doit être valide, et avoir été correctement ouvert par [fopen\(\)](#) ou [fsockopen\(\)](#).

[10.23.11 feof](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [feof](#) (int *fp*)

[feof\(\)](#) retourne TRUE si le pointeur *fp* est à la fin du fichier, ou si une erreur survient, sinon, retourne FALSE.

Le pointeur de fichier doit être valide, et avoir été correctement ouvert par [fopen\(\)](#), [popen\(\)](#), ou [fsockopen\(\)](#).

[10.23.12 fgetc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fgetc](#) (int *fp*)

[fgetc\(\)](#) retourne une chaîne contenant un seul caractère, lu depuis le fichier pointé par *fp*. Retourne FALSE à la fin du fichier (tout comme [feof\(\)](#)).

Le pointeur de fichier doit être valide, et avoir été correctement ouvert par [fopen\(\)](#), [popen\(\)](#), ou [fsockopen\(\)](#).

Voir aussi [fread\(\)](#), [fopen\(\)](#), [popen\(\)](#), [fsockopen\(\)](#), et [fgets\(\)](#).

[10.23.13 fgetcsv](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [fgetcsv](#)(int *fp*, int *length*, string *delimiter*)

Identique à [fgets\(\)](#) mais [fgetcsv\(\)](#) analyse la ligne qu'il lit et recherche les champs CSV, qu'il va retourner dans un tableau les contenant. Le délimiteur de champs *delimiter* est la virgule, à moins que vous ne fournissiez un troisième argument.

fp doit être un pointeur valide, et avoir été correctement ouvert par [fopen\(\)](#), [popen\(\)](#), ou [fsockopen\(\)](#). *length* doit être plus grand que la plus grande ligne trouvée dans un fichier CSV (en comptant les caractères de fin de ligne).

[fgetcsv\(\)](#) retourne FALSE en cas d'erreur, ou en cas de fin du fichier.

Note : une ligne vide dans un fichier CSV sera retournée dans le tableau comme une chaîne vide, et ne sera pas traitée comme une erreur.

exemple avec [fgetcsv\(\)](#) une ligne vide dans un fichier csv sera retournée dans le tableau comme une chaîne vide, et ne sera pas traitée comme une erreur. ***exemple avec [fgetcsv\(\)](#) une ligne vide dans un fichier csv sera retournée dans le tableau comme une chaîne vide, et ne sera pas traitée comme une erreur.***

```
$row = 1;
$fp = fopen ("test.csv", "r");
while ($data = fgetcsv ($fp, 1000, ",")) {
    $num = count ($data);
    print "<p> $num champs dans la ligne $row: <br>";
    $row++;
    for ($c=0; $c<$num; $c++) {
        print $data[$c] . " <br>";
    }
}
fclose ($fp);
```

[10.23.14 fgets](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fgets](#)(int *fp*, int *length*)

[fgets\(\)](#) retourne la chaîne lue jusqu'à la longueur *length* – 1 octets, ou bien la fin du fichier, ou encore un retour chariot (le premier des trois qui sera rencontré).

Si une erreur survient, retourne FALSE.

Erreur courante :

Les programmeurs habitués à la programmation 'C' noteront que [fgets\(\)](#) ne se comporte pas comme son équivalent C lors de la rencontre de la fin du fichier.

fp doit être valide, et avoir été correctement ouvert par [fopen\(\)](#), [popen\(\)](#), ou [fsockopen\(\)](#).

Un exemple simple : ***Lecture d'un fichier ligne par ligne***

```
$fd = fopen ("/tmp/inputfile.txt", "r");
while (!feof($fd)) {
```

```

    $buffer = fgets($fd, 4096);
echo $buffer;
}
fclose ($fd);

```

Voir aussi [fread\(\)](#), [fopen\(\)](#), [popen\(\)](#), [fgetc\(\)](#), et [fsockopen\(\)](#).

[10.23.15 fgetss](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fgetss](#) (int *fp*, int *length*, string *allowable_tags*)

Identique à [fgets\(\)](#), mais [fgetss\(\)](#) supprime toutes les balises HTML et PHP qu'il trouve dans le texte lu. Vous pouvez aussi préciser les balises qui seront ignorées dans le troisième paramètre, optionnel. Note : *allowable_tags* a été ajouté dans PHP 3.0.13, PHP4B3.

Voir aussi [fgets\(\)](#), [fopen\(\)](#), [fsockopen\(\)](#), [popen\(\)](#), et [strip_tags\(\)](#).

[10.23.16 file](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [file](#) (string *filename*, int *use_include_path*)

Identique à [readfile\(\)](#), hormis le fait que [file\(\)](#) retourne le fichier dans un tableau. Chaque élément du tableau correspond à une ligne du fichier, et les retour chariots sont placés en fin de ligne.

Vous pouvez utiliser l'option et en la mettant à "1", si vous voulez rechercher aussi dans le dossier [7.1.1.13 ini.include_path](#).

```

<?php
// Lire une page web dans un tableau, et l'afficher
$fcontents = file( 'http://www.php.net' );
while ( list( $line_num, $line ) = each( $fcontents ) ) {
echo "<b>Line $line_num:</b> " . htmlspecialchars( $line ) . "<br>\n";
}
// lire une page web dans une chaîne
$fcontents = join( '', file( 'http://www.php.net' ) );
?>

```

Voir aussi [readfile\(\)](#), [fopen\(\)](#) et [popen\(\)](#).

[10.23.17 file_exists](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [file_exists](#) (string *filename*)

Retourne TRUE si le fichier *filename* existe, et FALSE sinon.

[file_exists\(\)](#) ne fonctionne pas sur les fichiers distants. Les fichiers doivent être accessibles par le système de fichier du serveur.

Le résultat de cette fonction est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

[10.23.18 fileatime](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fileatime](#) (string *filename*)

[fileatime\(\)](#) renvoie la date à laquelle le fichier a été accédé pour la dernière fois, ou FALSE en cas d'erreur. Le résultat de cette fonction est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

[10.23.19 filectime](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filectime](#) (string *filename*)

[filectime\(\)](#) renvoie l'heure à laquelle l'inode *filename* a été accédé pour la dernière fois, ou FALSE en cas d'erreur.

Le résultat de cette fonction est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

[10.23.20 filegroup](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filegroup](#) (string *filename*)

[filegroup\(\)](#) renvoie le groupe qui possède le fichier *filename* ou FALSE en cas d'erreur. L'identifiant de groupe est retourné au format numérique, utilisez [posix_getgrgid\(\)](#) pour retrouver le nom du groupe.

Le résultat de cette fonction est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

Note : *Cette fonction est inopérante sous Windows*

[10.23.21 fileinode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fileinode](#) (string *filename*)

[fileinode\(\)](#) renvoie le numéro d'inode du fichier *filename*, ou FALSE en cas d'erreur.

Le résultat de cette fonction est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

Note : *Cette fonction est inopérante sous Windows*

[10.23.22 filemtime](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filemtime](#) (string *filename*)

[filemtime\(\)](#) renvoie la date de dernière modification du fichier *filename*, ou FALSE en cas d'erreur.

Le résultat de cette fonction est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

[10.23.23 fileowner](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fileowner](#)(string *filename*)

[fileowner\(\)](#) renvoie le nom du possesseur du fichier *filename*, ou FALSE en cas d'erreur. L'identification du possesseur de fichier est numérique : il faut utiliser [posix_getpwuid\(\)](#) pour retrouver le nom d'utilisateur. Le résultat de cette fonction est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

Note : *Cette fonction est inopérante sous Windows*

[10.23.24 fileperms](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fileperms](#)(string *filename*)

[fileperms\(\)](#) renvoie les permissions affectées au fichier *filename*, ou FALSE en cas d'erreur.

Le résultat de cette fonction est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

[10.23.25 filesize](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filesize](#)(string *filename*)

[filesize\(\)](#) renvoie la taille du fichier *filename*, ou FALSE en cas d'erreur.

Le résultat de cette fonction est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

[10.23.26 filetype](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [filetype](#)(string *filename*)

[filetype\(\)](#) renvoie le type du fichier *filename*. Les réponses possibles sont : fifo, char, dir, block, link, file, et unknown.

Retourne FALSE en cas d'erreur.

Le résultat de cette fonction est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

[10.23.27 flock](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [flock](#)(int *fp*, int *operation*)

PHP dispose d'un système complet de verrouillage de fichiers. Tous les programmes qui accèdent au fichier doivent utiliser la même méthode de verrouillage pour qu'il soit efficace.

[flock\(\)](#) agit sur le fichier *fp* qui doit avoir été ouvert au préalable. *operation* est une des valeurs suivantes :

- Acquisition d'un verrou : *operation* = 1.
- Acquisition d'un verrou exclusif (écriture), *operation* = 2.

- Libération d'un verrou (partagé ou exclusif), *operation* = 3.
- Si vous voulez que [flock\(\)](#) ne se bloque pas durant le verrouillage, ajoutez 4 à *operation*.

[flock\(\)](#) permet de réaliser un système de verrous écriture / lecture simple, qui peut être utilisé sur n'importe quelle plateforme (Unix et Windows compris).

[flock\(\)](#) retourne TRUE en cas de succès, et FALSE sinon. (le verrou n'a pas pu être obtenu).

Sur la plus part des OS, [flock\(\)](#) est implémenté au niveau processus. Lors de l'utilisation des API d'un serveur multi-thread, comme ISAPI, vous ne pouvez pas vous fier à [flock\(\)](#) pour protéger vos fichiers contre des accès concurrents du même serveur.

[10.23.28 fopen](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fopen](#) (string *filename*, string *mode*, int *use_include_path*)

Si *filename* commence par "http://" (insensible à la casse), une connexion HTTP 1.x est ouverte avec le serveur spécifié, et un pointeur sur la réponse fournie est retourné.

Attention, [fopen\(\)](#) ne gère pas les redirections, ce qui oblige à ajouter les slash " / " finaux pour indiquer un dossier.

Si *filename* commence par "ftp://" (insensible à la casse), une connexion FTP est ouverte avec le serveur spécifié, et un pointeur sur la réponse fournie est retourné. Si le serveur ne supporte pas le mode FTP passif, cette fonction échouera. Vous pouvez ouvrir des fichiers en lecture seulement, ou en écriture seulement (le full duplex n'est pas supporté).

Si *filename* commence par "php://stdin", "php://stdout", ou "php://stderr", le flot correspondant sera ouvert. (Cela a été introduit dans PHP 3.0.13; dans les anciennes versions, les fichiers "/dev/stdin" ou "/dev/fd/0" devaient être utilisés pour accéder à ces flots).

Si *filename* commence par n'importe quoi d'autre, PHP tentera de lire ce fichier dans le système local, et un pointeur sur le fichier ouvert sera retourné.

Si l'ouverture échoue, [fopen\(\)](#) retourne FALSE.

mode peut prendre les valeurs suivantes :

- 'r' – Ouvre en lecture seule, et place le pointeur de fichier au début du fichier.
- 'r+' – Ouvre en lecture et écriture, et place le pointeur de fichier au début du fichier.
- 'w' – Ouvre en écriture seule; place le pointeur de fichier au début du fichier et réduit la taille du fichier à 0. Si le fichier n'existe pas, on tente de le créer.
- 'w+' – Ouvre en lecture et écriture; place le pointeur de fichier au début du fichier et réduit la taille du fichier à 0. Si le fichier n'existe pas, on tente de le créer.
- 'a' – Ouvre en écriture seule; place le pointeur de fichier à la fin du fichier file. Si le fichier n'existe pas, on tente de le créer.
- 'a+' – Ouvre en lecture et écriture; place le pointeur de fichier à la fin du fichier. Si le fichier n'existe pas, on tente de le créer.

De plus, *mode* peut contenir la lettre 'b'. Cette option n'est utile que sur les systèmes qui font la différence entre les fichiers binaires et les fichiers textes (en bref, c'est inutile sous Unix). Si il n'est pas nécessaire, il sera ignoré.

Vous pouvez utiliser le troisième paramètre optionnel pour explorer le dossier [7.1.1.13 ini.include-path](#), en le mettant à 1.

Exemple avec fopen()

```
$fp = fopen("/home/rasmus/file.txt", "r");
$fp = fopen("http://www.php.net/", "r");
$fp = fopen("ftp://user:password@example.com/", "w");
```

Si vous rencontrez des problèmes en lecture ou écriture de fichier et que vous utilisez PHP en version module de serveur, n'oubliez pas que les fichiers auxquels vous accédez ne sont pas nécessairement accessibles au processus serveur.

Sous Windows, assurez vous de bien échapper les backslash utilisés dans le chemin du fichier, ou bien utilisez des slash.

```
$fp = fopen("c:\\data\\info.txt", "r");
```

Voir aussi [fclose\(\)](#), [fsockopen\(\)](#) et [popen\(\)](#).

10.23.29 fpassthru

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fpassthru](#) (int *fp*)

[fpassthru\(\)](#) lit tout le reste d'un fichier jusqu'à la fin, et dirige le résultat vers la sortie standard.

Si une erreur survient, [fpassthru\(\)](#) retourne FALSE.

Le pointeur de fichier doit être valide, et doit avoir été correctement ouvert par [fopen\(\)](#), [popen\(\)](#), ou [fsockopen\(\)](#). Après lecture, [fpassthru\(\)](#) va fermer le fichier (le pointeur *fp* sera alors invalide).

Si vous voulez simplement afficher le contenu d'un fichier, il suffit d'utiliser [readfile\(\)](#), ce qui épargnera l'appel à [fopen\(\)](#).

Voir aussi [readfile\(\)](#), [fopen\(\)](#), [popen\(\)](#), et [fsockopen\(\)](#).

10.23.30 fputs

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fputs](#) (int *fp*, string *str*, int *length*)

[fputs\(\)](#) est un alias de [fwrite\(\)](#), et lui est identique en tout point. Notez que *length* est un paramètre optionnel, et si il n'est pas spécifié, toute la chaîne est écrite.

10.23.31 fread

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fread](#) (int *fp*, int *length*)

[fread\(\)](#) lit jusqu'à *length* octets dans le fichier référencé par *fp*. La lecture s'arrête lorsque *length* octets ont été lus, ou que l'on a atteint la fin du fichier, ou qu'une erreur survient (le premier des trois).

```
// Lit un fichier, et le place dans une chaîne
$filename = "/usr/local/something.txt";
$fd = fopen ($filename, "r");
```

```
$contents = fread ($fd, filesize ($filename));
fclose ($fd);
```

Voir aussi [fwrite\(\)](#), [fopen\(\)](#), [fsockopen\(\)](#), [popen\(\)](#), [fgets\(\)](#), [fgetss\(\)](#), [file\(\)](#), et [fpassthru\(\)](#).

10.23.32 fscanf

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [fscanf](#)(int *handle*, string *format*, string *var1*...)

[fscanf\(\)](#) est similaire à [@xref{function.sscanf , , sscanf\(\)}](#), mais elle prend comme entrée un fichier, associé à *handle* et l'interprète en fonction du format *format*. Si seulement deux paramètres sont passés à la fonction, les valeurs parsées seront retournées sous forme de tableau. Si des arguments optionnels sont passés, la fonction retournera le nombre de valeurs assignées. Les options doivent être passées par référence. *Exemple [fscanf\(\)](#)*

```
$fp = fopen ("users.txt", "r");
while ($userinfo = fscanf ($fp, "%s\t%s\t%s\n")) {
list ($name, $profession, $countrycode) = $userinfo;
    //... traitement des données
}
fclose($fp);
```

users.txt

```
janus  argonaute      gr
rodin  sculpteur       fr
som    oncle us
leonard inventeur it
```

Voir aussi [fread\(\)](#), [fgets\(\)](#), [fgetss\(\)](#), [@xref{function.sscanf , , sscanf\(\)}](#), [printf\(\)](#), et [sprintf\(\)](#).

10.23.33 fseek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fseek](#)(int *fp*, int *offset*)

[fseek\(\)](#) positionne le pointeur interne de fichier dans le fichier référencé par *fp* à l'offset *offset*. Equivalent à la fonction C `fseek(fp, offset, SEEK_SET)`.

Retourne TRUE en cas de succès, et sinon -1. Notez que positionner le pointeur au delà de la fin du fichier n'est pas une erreur.

Ne peut pas être utilisé sur les pointeurs retournés par [fopen\(\)](#) si ils sont au format HTTP ou FTP.

Voir aussi [ftell\(\)](#) et [rewind\(\)](#).

10.23.34 fstat

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [fstat](#) (int *fp*)

Rassemble les informations sur le fichier dont on connaît le pointeur *fp*. Cette fonction est similaire à la fonction [stat\(\)](#), hormis le fait qu'elle utilise un pointeur de fichier, au lieu d'un nom de fichier.

Retourne un tableau avec les éléments suivants :

1. volume
2. inode
3. mode de protection du inode
4. nombre de liens
5. id de l'utilisateur propriétaire
6. id du groupe propriétaire
7. type du volume de l' inode *
8. taille en octets
9. date du dernier accès
10. date de la dernière modification
11. date du dernier changement
12. taille de bloc du système pour les entrées/sorties *
13. Nombre de blocs alloués

* – uniquement sur les systèmes qui supporte le type `st_blksize` type§les autres systèmes (i.e. Windows) retourne -1

Les résultats de cette fonction sont mis en cache . Reportez vous à la fonction [clearstatcache\(\)](#) pour plus de détails.

10.23.35 ftell

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftell](#) (int *fp*)

[ftell\(\)](#) retourne la position courante du pointeur dans le fichier repéré par le pointeur *fp*, i.e., son offset. Si une erreur survient, retourne FALSE.

Le pointeur de fichier doit être valide, et avoir été correctement ouvert par [fopen\(\)](#) ou [popen\(\)](#).

Voir aussi [fopen\(\)](#), [popen\(\)](#), [fseek\(\)](#) et [rewind\(\)](#).

10.23.36 ftruncate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftruncate](#) (int *fp*, int *size*)

[ftruncate\(\)](#) prend le pointeur de fichier *fp* et le tronque à la taille de *size*. La fonction retourne TRUE en cas de succès, et false sinon.

[10.23.37 fwrite](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fwrite](#)(int *fp*, string *string*, int *length*)

[fwrite\(\)](#) écrit le contenu de la chaîne *string* dans le fichier pointé par *fp*. Si la longueur *length* est fournie, l'écriture s'arrêtera après *length* octets, ou à la fin de la chaîne (le premier des deux).

Notez que si *length* est fourni, alors l'option de configuration [7.1.1.17 ini.magic-quotes-runtime](#) sera ignorée, et les slash seront conservés.

Voir aussi [fread\(\)](#), [fopen\(\)](#), [fsockopen\(\)](#), [popen\(\)](#), et [fputs\(\)](#).

[10.23.38 set file buffer](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [set_file_buffer](#)(int *fp*, int *buffer*)

L'écriture de fichier avec [fwrite\(\)](#) utilise normalement un buffer de 8K. Cela signifie que si deux processus essaie d'écrire dans le même fichier, ils font une pause tous les 8ko pour laisser le temps à l'autre d'écrire à son tour. [set_file_buffer\(\)](#) permet de modifier la taille du buffer de sortie pour le pointeur de fichier *fp* à *buffer* octets. Si *buffer* vaut 0, l'écriture se fera sans buffer. Cela force l'écriture de toutes les données par un processus avant que les autres puisse accéder au fichier.

La fonction retourne 0 en cas de succès, ou EOF si la requête ne peut pas être honorée.

L'exemple suivant montre comment utiliser la fonction [set_file_buffer\(\)](#) pour créer un fichier sans buffer.

Exemple avec [set_file_buffer\(\)](#)

```
$fp=fopen($file, "w");
if($fp){
set_file_buffer($fp, 0);
fputs($fp, $output);
fclose($fp);
}
```

Voir aussi [fopen\(\)](#), [fwrite\(\)](#).

[10.23.39 is_dir](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_dir](#)(string *filename*)

Retourne TRUE si *filename* existe et est un dossier.

Le résultat de cette fonction est mis en cache. Voir la fonction [clearstatcache\(\)](#) pour plus de détails.

Voir aussi [is_file\(\)](#) et [is_link\(\)](#).

[10.23.40 is_executable](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_executable](#)(string *filename*)

Retourne TRUE si *filename* existe et est exécutable.

Le résultat de cette fonction est mis en cache. Voir la fonction [clearstatcache\(\)](#) pour plus de détails.

Voir aussi [is_file\(\)](#) et [is_link\(\)](#).

[10.23.41 is_file](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_file](#)(string *filename*)

Retourne TRUE si *filename* existe et est un fichier (et pas un dossier).

Le résultat de cette fonction est mis en cache. Voir la fonction [clearstatcache\(\)](#) pour plus de détails.

Voir aussi [is_dir\(\)](#) et [is_link\(\)](#).

[10.23.42 is_link](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_link](#)(string *filename*)

Retourne TRUE si *filename* existe et est un lien symbolique.

Le résultat de cette fonction est mis en cache. Voir la fonction [clearstatcache\(\)](#) pour plus de détails.

Voir aussi [is_dir\(\)](#) et [is_file\(\)](#).

Note : *Cette fonction est inopérante sous Windows*

[10.23.43 is_readable](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_readable](#)(string *filename*)

[is_readable\(\)](#) retourne vrai si *filename* existe et est accessible en lecture.

N'oubliez pas que PHP accède aux fichiers avec les mêmes autorisations que l'utilisateur qui fait tourner le serveur web (souvent, c'est 'nobody', personne).

Le résultat de cette fonction est mis en cache. Voir la fonction [clearstatcache\(\)](#) pour plus de détails.

Voir aussi [is_writeable\(\)](#).

[10.23.44 is_writeable](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_writeable](#)(string *filename*)

[is_writeable\(\)](#) retourne TRUE si *filename* existe et est accessible en lecture.

N'oubliez pas que PHP accède aux fichiers avec les mêmes autorisations que l'utilisateur qui fait tourner le serveur web (souvent, c'est 'nobody', personne).

Le résultat de cette fonction est mis en cache. Voir la fonction [clearstatcache\(\)](#) pour plus de détails.

Voir aussi [is_readable\(\)](#).

10.23.45 is_uploaded_file

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_uploaded_file](#) (string *filename*)

Cette fonction est disponible à partir des version PHP 3.0.16 et 4.0.2.

Retourne true si le fichier *filename* a été téléchargé par HTTP POST. Cela est très utile pour vous assurer qu'un utilisateur n'essaie pas d'accéder intentionnellement à un fichier auquel il n'a pas droit (comme `~/etc/passwd`).

Ce type de vérification est spécialement important si il est possible que les fichiers téléchargés révèlent leur contenu à l'utilisateur, ou même aux utilisateurs du même système.

Voir aussi [move_uploaded_file\(\)](#), et la section [8.4 Gestion des chargements de fichier](#) pour un exemple simple.

10.23.46 link

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [link](#) (string *target*, string *link*)

[link\(\)](#) crée un lien.

Voir aussi [symlink\(\)](#) pour créer des liens symboliques et [readlink\(\)](#) avec [linkinfo\(\)](#).

Note : *Cette fonction est inopérante sous Windows*

10.23.47 linkinfo

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [linkinfo](#) (string *path*)

[linkinfo\(\)](#) renvoie les informations à propos d'un lien le champs `st_dev` de la structure d'information d'UNIX (comme en langage C). Cette fonction sert à vérifier si un lien (repéré par *path*) existe (en utilisant la même méthode que la macro `S_ISLNK` de `stat.h`). Retourne FALSE en cas d'erreur.

Voir aussi [symlink\(\)](#), [link\(\)](#), et [readlink\(\)](#).

Note : *Cette fonction est inopérante sous Windows*

10.23.48 mkdir

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mkdir](#) (string *pathname*, int *mode*)

Tente de créer un dossier dans le chemin *pathname*.

Notez que vous aurez à préciser le mode en base octale, ce qui signifie que vous aurez probablement un 0 comme premier chiffre :

```
mkdir ("/path/to/my/dir", 0700);
```

Retourne TRUE en cas de succès, et FALSE en cas d'échec.

Voir aussi [rmdir\(\)](#).

[10.23.49 move_uploaded_file](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [move_uploaded_file](#)(string *filename*, string *destination*)

Cette fonction est disponible à partir des version PHP 3.0.16 et 4.0.2.

Cette fonction s'assure que le fichier *filename* est un fichier téléchargé par HTTP POST. Si le fichier est valide, il est déplacé jusqu'à *destination*.

Si *filename* n'est pas valide, rien ne se passe, et [move_uploaded_file\(\)](#) retournera false.

Si *filename* est un fichier téléchargé, mais que pour une raison quelconque, il ne peut être déplacé, rien ne se passe, et [move_uploaded_file\(\)](#) retourne false. De plus, une alerte sera affichée.

Ce type de vérification est spécialement important si il est possible que les fichiers téléchargés révèlent leur contenu à l'utilisateur, ou même aux utilisateurs du même système.

Voir aussi [move_uploaded_file\(\)](#), et la section [8.4 Gestion des chargements de fichier](#) pour un exemple simple.

[10.23.50 pclose](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pclose](#)(int *fp*)

[pclose\(\)](#) ferme un processus de pointeur de fichier ouvert par [popen\(\)](#).

Le pointeur de fichier doit être valide, et avoir été ouvert correctement par [popen\(\)](#).

Retourne le statut final du processus exécuté.

Voir aussi [popen\(\)](#).

[10.23.51 popen](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [popen](#)(string *command*, string *mode*)

[popen\(\)](#) ouvre un processus fils en faisant un fork de la commande.

[popen\(\)](#) retourne un pointeur de fichier identique à celui retourné par [fopen\(\)](#), hormis le fait qu'il sera unidirectionel (lecture seule, ou écriture seule), et doit être terminé par [pclose\(\)](#). Ce pointeur peut être utilisé avec [fgets\(\)](#), [fgetss\(\)](#), et [fputs\(\)](#).

Si une erreur survient, retourne FALSE.

```
$fp = popen ("/bin/ls", "r");
```

Voir aussi [pclose\(\)](#).

[10.23.52 readfile](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [readfile](#)(string *filename*, int *use_include_path*)

[readfile\(\)](#) lis le fichier *filename* et l'envoi à la sortie standard.

Retourne le nombre d'octets lus depuis le fichier. Si une erreur survient, retourne FALSE.

Si *filename* commence par "http://" (insensible à la casse), une connexion HTTP 1.0 sera ouverte avec le serveur spécifié, et le texte de la réponse sera affiché sur la sortie standard.

ATTENTION : PHP ne gère pas les redirections, donc il faut penser à ajouter un "/" final pour les dossiers.

Si *filename* commence par "ftp://" (insensible à la casse), une connexion FTP est ouverte avec l'hôte spécifié et la réponse du serveur est affichée. Si le serveur ne supporte les connexions passives, la requête échouera.

Si *filename* ne commence par aucun des cas précédents, le fichier sera ouvert sur l'hôte local, et envoyé à la sortie standard.

Vous pouvez utiliser le deuxième paramètre optionnel pour explorer le dossier [7.1.1.13 ini.include-path](#), en passant la valeur de 1.

Voir aussi [fpasssthru\(\)](#), [file\(\)](#), [fopen\(\)](#), [include\(\)](#), [require\(\)](#), et [virtual\(\)](#).

[10.23.53 readlink](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [readlink](#) (string *path*)

[readlink\(\)](#) fait la même chose que la fonction readlink en C : elle retourne le contenu du lien symbolique repéré par *path*, ou FALSE en cas d'erreur.

Voir aussi [symlink\(\)](#), [readlink\(\)](#) et [linkinfo\(\)](#).

Note : Cette fonction est inopérante sous Windows

[10.23.54 rename](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [rename](#) (string *oldname*, string *newname*)

[rename\(\)](#) tente de renommer le fichier *oldname* en *newname*.

Retourne TRUE en cas de succès et FALSE sinon.

[10.23.55 rewind](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [rewind](#) (int *fp*)

Remplace le pointeur du fichier *fp* au début.

Si une erreur survient, retourne FALSE.

Le pointeur de fichier doit être valide, et avoir été correctement ouvert par [fopen\(\)](#).

Voir aussi [fseek\(\)](#) et [ftell\(\)](#).

[10.23.56 rmdir](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [rmdir](#) (string *dirname*)

[rmdir\(\)](#) tente d'effacer le dossier dont le chemin est *dirname*. Le dossier doit être vide, et le script doit avoir les autorisations adéquates.

Si une erreur survient, retourne FALSE.

Voir aussi [mkdir\(\)](#).

10.23.57 stat

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [stat](#)(string *filename*)

[stat\(\)](#) renvoie les informations à propos du fichier filename.

Retourne un tableau avec les éléments suivants :

1. volume
2. inode
3. mode de protection du inode
4. nombre de liens
5. id de l'utilisateur propriétaire
6. id du groupe propriétaire
7. type du volume de l' inode *
8. taille en octets
9. date du dernier accès
10. date de la dernière modification
11. date du dernier changement
12. taille de bloc du système pour les entrées/sorties *
13. Nombre de blocs alloués

* – uniquement sur les systèmes qui supporte le type st_blksize type§les autres systèmes (i.e. Windows) retourne -1

Les résultats de cette fonction sont mis en cache . Reportez vous à la fonction [clearstatcache\(\)](#) pour plus de détails.

10.23.58 lstat

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [lstat](#)(string *filename*)

Cette fonction est identique à [stat\(\)](#) mais elle accepte aussi un lien symbolique comme argument.

Retourne un tableau avec les éléments suivants :

1. volume
2. inode
3. mode de protection du inode
4. nombre de liens
5. id de l'utilisateur propriétaire
6. id du groupe propriétaire
7. type du volume de l' inode *
8. taille en octets
9. date du dernier accès
10. date de la dernière modification
11. date du dernier changement
12. taille de bloc du système pour les entrées/sorties *

13. Nombre de blocs alloués

* – uniquement sur les systèmes qui supporte le type `st_blksize` type§les autres systèmes (i.e. Windows) retourne -1

Les résultats de cette fonction sont mis en cache . Reportez vous à la fonction [clearstatcache\(\)](#) pour plus de détails.

10.23.59 realpath

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [realpath](#) (string *path*)

[realpath\(\)](#) résoud tous les liens symbolique, et remplace toutes les références './.', './..' et '/' de *path* puis retourne le chemin canonique absolu ainsi trouvé. Le résultat ne contient aucun lien symbolique, './.' ou './..'.
Exemple [realpath\(\)](#)

```
$real_path = realpath ("../../index.php");
```

10.23.60 symlink

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [symlink](#) (string *target*, string *link*)

[symlink\(\)](#) crée un lien symbolique pour l'objet *target* avec le nom de *link*. Voir aussi [link\(\)](#) pour créer des liens durs, et [readlink\(\)](#) ainsi que [linkinfo\(\)](#).

Note : *Cette fonction est inopérante sous Windows*

10.23.61 tempnam

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [tempnam](#) (string *dir*, string *prefix*)

[tempnam\(\)](#) crée un fichier temporaire unique dans le dossier *dir*. Si le dossier n'existe pas, [tempnam\(\)](#) va générer un nom de fichier dans le dossier temporaire du système.

Le comportement de [tempnam\(\)](#) dépend du système. Sous Windows, la variable d'environnement TMP sera prioritaire par rapport au paramètre *dir*; sous Linux la variable d'environnement TMPDIR aura la priorité, tandis que l'OS SVR4 utilisera toujours le paramètre *dir* si le dossier n'existe pas. Reportez vous à la documentation de votre système([tempnam\(3\)](#)).

Retourne le nom du fichier temporaire, ou la chaîne NULL en cas d'échec. *Exemple avec [tempnam\(\)](#)*

```
$tmpfname = tempnam ("/tmp", "FOO");
```

[10.23.62 tmpfile](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [tmpfile](#)(void)

[tmpfile\(\)](#) crée un fichier temporaire avec un nom unique, ouvert en écriture, et retourne un pointeur de fichier, identique à ceux retourné par [fopen\(\)](#). Ce fichier sera automatiquement effacé lorsqu'il sera fermé (avec [fclose\(\)](#)), ou lorsque le fichier sera terminé.

Pour plus de détails, consultez votre documentation système sur la fonction tmpfile(3), et sur ``stdio.h'`. Voir aussi [tempnam\(\)](#).

[10.23.63 touch](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [touch](#)(string *filename*, int *time*)

[touch\(\)](#) tente de forcer la date de modification du fichier nommé *filename* à la date de *time*. Si *time* est omis, c'est l'heure courante qui est utilisée.

Si le fichier n'existe pas, il est créé.

Retourne TRUE en cas de succès, et FALSE sinon. *Exemple avec [touch\(\)](#)*

```
if ( touch($FileName) ) {
print "$FileName modification time has been changed to todays date et time";
} else {
print "Sorry Could Not change modification time of $FileName";
}
```

[10.23.64 umask](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [umask](#)(int *mask*)

[umask\(\)](#) change le umask de PHP : `mask & 0777` et retourne le vieux umask. Lorsque PHP est utilisé comme module de serveur, le umask reprend sa valeur à la fin de chaque script.

[umask\(\)](#) appelé sans argument retourne simplement le umask courant.

[10.23.65 unlink](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [unlink](#)(string *filename*)

[unlink\(\)](#) efface *filename*. Identique à la fonction Unix C `unlink()`.

Retourne FALSE en cas d'échec.

Voir aussi [rmdir\(\)](#) pour supprimer des dossiers.

Note : *Cette fonction peut ne pas fonctionner sous Windows.*

10.24 FTP

[\[Notes en ligne\]](#)

FTP : File Transfer Protocol.

Les constantes suivantes sont définies dans le module FTP : FTP_ASCII, et FTP_BINARY.

10.24.1 ftp_connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_connect](#)(string *host*, int *port*)

Retourne un flot FTP en cas de succès, et FALSE sinon.

[ftp_connect\(\)](#) ouvre une connexion FTP avec l'hôte *host*. Le paramètre *port* spécifie le port de connexion. Si il est omis, le port 21 sera utilisé.

10.24.2 ftp_login

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_login](#)(int *ftp_stream*, string *username*, string *password*)

Retourne TRUE en cas de succès, et FALSE sinon.

Authentifie le flot FTP.

10.24.3 ftp_pwd

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_pwd](#)(int *ftp_stream*)

Retourne le nom du dossier courant, ou FALSE en cas d'erreur.

10.24.4 ftp_cdup

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_cdup](#)(int *ftp_stream*)

Retourne TRUE en cas de succès, et FALSE sinon.

Change de dossier, et passe au dossier parent.

10.24.5 ftp_chdir

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_chdir](#)(int *ftp_stream*, string *directory*)

Retourne TRUE en cas de succès, et FALSE sinon.

Change le dossier courant en *directory*.

[10.24.6 ftp_mkdir](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ftp_mkdir](#) (int *ftp_stream*, string *directory*)

Retourne le nom du dossier ainsi créé en cas de succès, et FALSE sinon.
Crée le dossier nommé *directory*.

[10.24.7 ftp_rmdir](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_rmdir](#) (int *ftp_stream*, string *directory*)

Retourne TRUE en cas de succès, et FALSE sinon.
Efface le dossier *directory*.

[10.24.8 ftp_nlist](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_nlist](#) (int *ftp_stream*, string *directory*)

Retourne un tableau de nom de fichiers en cas de succès, et FALSE sinon.

[10.24.9 ftp_rawlist](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_rawlist](#) (int *ftp_stream*, string *directory*)

[ftp_rawlist\(\)](#) exécute la commande FTP LIST, et retourne le résultat dans un tableau. Chaque élément du tableau correspond à une ligne du résultat de la commande. Le résultat n'est pas analysé, et est retourné brut. L'identifiant de système retourné par [ftp_systype\(\)](#) sera utile pour déterminer la façon d'interpréter le résultat.

[10.24.10 ftp_systype](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_systype](#) (int *ftp_stream*)

Retourne le type de serveur, ou FALSE en cas d'erreur.

[10.24.11 ftp_pasv](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_pasv](#) (int *ftp_stream*, int *pasv*)

Retourne TRUE en cas de succès, et FALSE sinon.
[ftp_pasv\(\)](#) active le mode passif si *pasv* est à TRUE (et le désactive si *pasv* est à FALSE.) En mode passif, les

données de connexion sont initiées par le client, plutôt que par le serveur.

[10.24.12 ftp_get](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_get](#)(int *ftp_stream*, string *local_file*, string *remote_file*, int *mode*)

Retourne TRUE en cas de succès, et FALSE sinon.

[ftp_get\(\)](#) télécharge le fichier *remote_file* depuis le serveur FTP, et le sauve dans le fichier local *local_file*. Le mode de transfert *mode* spécifié doit être soit FTP_ASCII ou FTP_BINARY.

[10.24.13 ftp_fget](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_fget](#)(int *ftp_stream*, int *fp*, string *remote_file*, int *mode*)

Retourne TRUE en cas de succès, et FALSE sinon.

[ftp_fget\(\)](#) télécharge le fichier *remote_file* depuis le serveur FTP, et l'écrit dans le fichier identifié par *fp*. Le mode de transfert *mode* spécifié doit être FTP_ASCII ou FTP_BINARY.

[10.24.14 ftp_put](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_put](#)(int *ftp_stream*, string *remote_file*, string *local_file*, int *mode*)

Retourne TRUE en cas de succès, et FALSE sinon.

[ftp_put\(\)](#) enregistre le fichier *local_file* sur le serveur FTP, sous le nom de *remote_file*. Le mode de transfert *mode* spécifié doit être FTP_ASCII ou FTP_BINARY.

[10.24.15 ftp_fput](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_fput](#)(int *ftp_stream*, string *remote_file*, int *fp*, int *mode*)

Retourne TRUE en cas de succès, et FALSE sinon.

[ftp_fput\(\)](#) charge les données issues du fichier identifié par *fp* jusqu'à la fin du fichier. Le résultat est stocké dans le fichier *remote_file* sur le serveur FTP. Le mode de transfert *mode* spécifié doit être FTP_ASCII ou FTP_BINARY.

[10.24.16 ftp_size](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_size](#)(int *ftp_stream*, string *remote_file*)

Retourne la taille du fichier en cas de succès, et FALSE sinon.

[ftp_size\(\)](#) retourne la taille d'un fichier sur un serveur FTP. Si une erreur survient, ou que le fichier n'existe pas, la valeur -1 est retournée. Certains serveurs FTP ne supportent pas cette fonction.

[10.24.17 ftp_mdtm](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_mdtm](#)(int *ftp_stream*, string *remote_file*)

Retourne un UNIX timestamp en cas de succès, et FALSE sinon.

[ftp_mdtm\(\)](#) lit la date de dernière modification d'un fichier et retourne le UNIX timestamp. Si une erreur survient, ou si le fichier n'existe pas, la valeur -1 est retournée. Certains serveurs FTP ne supportent pas cette fonction.

[10.24.18 ftp_rename](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_rename](#)(int *ftp_stream*, string *from*, string *to*)

Retourne TRUE en cas de succès, et FALSE sinon.

[ftp_rename\(\)](#) renomme le fichier *from* en *to*.

[10.24.19 ftp_delete](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_delete](#)(int *ftp_stream*, string *path*)

Retourne TRUE en cas de succès, et FALSE sinon.

[ftp_delete\(\)](#) efface le fichier *path* sur un serveur FTP.

[10.24.20 ftp_site](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_site](#)(int *ftp_stream*, string *cmd*)

Retourne TRUE en cas de succès, et FALSE sinon.

[ftp_site\(\)](#) envoie la commande *cmd* au serveur FTP. Les commandes SITE ne sont pas normalisées, et peuvent varier d'un serveur à l'autre. Elles permettent de gérer notamment les permissions de fichier, et les groupes.

[10.24.21 ftp_quit](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_quit](#)(int *ftp_stream*)

[ftp_connect\(\)](#) ferme la connexion *ftp_stream*.

[10.25 Function Handling functions](#)

[\[Notes en ligne\]](#)

These functions all handle various operations involved in working with functions.

10.25.1 call_user_func

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [call_user_func](#)(string *function_name* , mixed *parameter* , mixed ...)

Call a user defined function given by the *function_name* parameter. Take the following:

```
function barber ($type) {
print "You wanted a $type haircut, no problem";
}
call_user_func ('barber', "mushroom");
call_user_func ('barber', "shave");
```

10.25.2 create_function

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [create_function](#)(string *args*, string *code*)

Creates an anonymous function from the parameters passed, and returns a unique name for it. Usually the *args* will be passed as a single quote delimited string, and this is also recommended for the *code*. The reason for using single quoted strings, is to protect the variable names from parsing, otherwise, if you use double quotes there will be a need to escape the variable names, e.g. `\$avar`.

You can use this function, to (for example) create a function from information gathered at run time: creating an anonymous function with [create_function\(\)](#) *creating an anonymous function with create_function()*

```
$newfunc = create_function('$a,$b','return "ln($a) + ln($b) = ".log($a * $b);');
echo "New anonymous function: $newfunc\n";
echo $newfunc(2,M_E)."\n";
// outputs
// New anonymous function: lambda_1
// ln(2) + ln(2.718281828459) = 1.6931471805599
```

Or, perhaps to have general handler function that can apply a set of operations to a list of parameters: making a general processing function with [create_function\(\)](#) *making a general processing function with create_function()*

```
function process($var1, $var2, $farr) {
for ($f=0; $f < count($farr); $f++)
echo $farr[$f]($var1,$var2)."\n";
}
// create a bunch of math functions
$f1 = 'if ($a >=0) {return "b*a^2 = ".$b*sqrt($a);} else {return false;}';
$f2 = "return \"min(b^2+a, a^2,b) = \".min(\$a*\$a+\$b,\$b*\$b+\$a);";
$f3 = 'if ($a > 0 && $b != 0) {return "ln(a)/b = ".log($a)/$b;} else {return false;}';
$farr = array(
```

```

create_function('$x,$y', 'return "some trig: ".(sin($x) + $x*cos($y));'),
create_function('$x,$y', 'return "a hypotenuse: ".sqrt($x*$x + $y*$y);'),
create_function('$a,$b', $f1),
create_function('$a,$b', $f2),
create_function('$a,$b', $f3)
);
echo "\nUsing the first array of anonymous functions\n";
echo "parameters: 2.3445, M_PI\n";
process(2.3445, M_PI, $farr);
// now make a bunch of string processing functions
$garr = array(
create_function('$b,$a', 'if (strcmp($a,$b,3) == 0) return "** \"$a\" ' .
    'and \"$b\" \n** Look the same to me! (looking at the first 3 chars)";'),
create_function('$a,$b', '; return "CRCs: ".crc32($a)." , ".crc32(b);'),
create_function('$a,$b', '; return "similar(a,b) = ".similar_text($a,$b,p).("($p%)"');')
);
echo "\nUsing the second array of anonymous functions\n";
process("Twas brillling and the slithy toves", "Twas the night", $garr);

```

and when you run the code above, the output will be:

```

Using the first array of anonymous functions
parameters: 2.3445, M_PI
some trig: -1.6291725057799
a hypotenuse: 3.9199852871011
b*a^2 = 4.8103313314525
min(b^2+a, a^2,b) = 8.6382729035898
ln(a/b) = 0.27122299212594
Using the second array of anonymous functions
** "Twas the night" and "Twas brillling and the slithy toves"
** Look the same to me! (looking at the first 3 chars)
CRCs: -725381282 , 1908338681
similar(a,b) = 11(45.833333333333%)

```

But perhaps the most common use for of lambda-style (anonymous) functions is to create callback functions, for example when using [array_walk\(\)](#) or [usort\(\)](#) *Using anonymous functions as callback functions*

```

$av = array("the ", "a ", "that ", "this ");
array_walk($av, create_function('v,$k', '$v = $v."mango";'));
print_r($av); // for PHP 3 use var_dump()
// outputs:
// Array
// (
//     [0] => the mango
//     [1] => a mango
//     [2] => that mango
//     [3] => this mango
// )
// an array of strings ordered from shorter to longer
$sv = array("small", "larger", "a big string", "it is a string thing");
print_r($sv);
// outputs:
// Array
// (
//     [0] => small
//     [1] => larger
//     [2] => a big string
//     [3] => it is a string thing

```

```
// )
// sort it from longer to shorter
usort($sv, create_function('$a,$b','return strlen($b) - strlen($a);'));
print_r($sv);
// outputs:
// Array
// (
//     [0] => it is a string thing
//     [1] => a big string
//     [2] => larger
//     [3] => small
// )
```

[10.25.3 func_get_arg](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [func_get_arg](#) (int *arg_num*)

Retourne l'argument à la *arg_num*-ième position d'une fonction définie par l'utilisateur. Les positions sont indexées en commençant à zéro. [func_get_arg\(\)](#) générera une alerte (warning) si elle est appelée hors d'une définition de fonction.

Si *arg_num* est plus grand que le nombre d'arguments une alerte sera générée, et [func_get_arg\(\)](#) retournera FALSE.

```
<?php
function foo() {
    $numargs = func_num_args();
    echo "Nombre d'arguments: $numargs<br>\n";
    if ($numargs >= 2) {
        echo "Deuxième argument : " . func_get_arg (1) . "<br>\n";
    }
}
foo (1, 2, 3);
?>
```

[func_get_arg\(\)](#) peut être utilisé conjointement avec [func_num_args\(\)](#) et [func_get_args\(\)](#) pour permettre aux fonctions définies par les utilisateurs d'accepter un nombre variable d'argument.

Note : Cette fonction a été ajoutée dans PHP 4.

[10.25.4 func_get_args](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [func_get_args](#) (void)

Retourne un tableau dont les éléments sont les arguments de la fonction courante. [func_get_args\(\)](#) générera une alerte (warning) si elle est appelée hors d'une définition de fonction.

```
<?php
function foo() {
    $numargs = func_num_args();
    echo "Nombre d'arguments: $numargs<br>\n";
```

```

if ($numargs >= 2) {
echo "Deuxième argument : " . func_get_arg (1) . "<br>\n";
}
$args_list = func_get_args();
for ($i = 0; $i < $numargs; $i++) {
echo "Argument $i : " . $args_list[$i] . "<br>\n";
}
}
foo (1, 2, 3);
?>

```

[func_get_args\(\)](#) peut être utilisé conjointement avec [func_num_args\(\)](#) et [func_get_arg\(\)](#) pour permettre aux fonctions définies par les utilisateurs d'accepter un nombre variable d'argument.

Note : Cette fonction a été ajoutée dans PHP 4.

10.25.5 func_num_args

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [func_num_args](#) (void)

Retourne le nombre d'arguments passé à la fonction courante. [func_num_args\(\)](#) génèrera une alerte (warning) si elle est appelée hors d'une définition de fonction.

```

<?php
function foo() {
    $numargs = func_num_args();
    echo "Nombre d'arguments: $numargs\n";
}
foo (1, 2, 3); // affiche : 'Number d'arguments: 3'
?>

```

[func_num_args\(\)](#) peut être utilisé conjointement avec [func_get_arg\(\)](#) et [func_get_args\(\)](#) pour permettre aux fonctions définies par les utilisateurs d'accepter un nombre variable d'arguments.

Note : Cette fonction a été ajoutée dans PHP 4.

10.25.6 function_exists

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [function_exists](#) (string *function_name*)

Vérifie la liste des fonctions définies, et recherche la fonction *function_name*. Retourne TRUE si cette fonction est trouvée, sinon FALSE.

10.25.7 register_shutdown_function

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [register_shutdown_function](#) (string *func*)

Enregistre la fonction *func* pour exécution à la fin du script.

Erreur commune :

Etant donné qu'aucun affichage n'est autorisé depuis cette fonction, vous ne pourrez pas déboguer en utilisant `print` ou `echo` dans cette fonction.

10.26 Fonctions GNU Gettext

[\[Notes en ligne\]](#)

10.26.1 bindtextdomain

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bindtextdomain](#) (string *domain*, string *directory*)

[bindtextdomain\(\)](#) fixe le chemin du domaine *domain* à *directory*.

10.26.2 dcgettext

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dcgettext](#) (string *domain*, string *message*, int *category*)

[dcgettext\(\)](#) permet de remplacer le domaine courant lors de la recherche d'un message. Elle permet aussi de spécifier une catégorie.

10.26.3 dgettext

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dgettext](#) (string *domain*, string *message*)

[dgettext\(\)](#) remplace le domaine courant.

10.26.4 gettext

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gettext](#) (string *message*)

[gettext\(\)](#) retourne une chaîne traduite, si elle en a trouvé une dans la table de traduction, ou bien le message *message*, s'il n'a pas été trouvé. Vous pouvez utiliser le caractère souligné (_) comme alias de cette fonction.

Vérification [gettext\(\)](#)

```
<?php
// Choix l'allemand
putenv ("LANG=de");
// Spécifie la localisation des tables de traduction
bindtextdomain ("myPHPApp", "./locale");
// Choisi le domaine
textdomain ("myPHPApp");
// Affiche un message de test
print (gettext ("Bienvenue sur mon application PHP"));
```

?>

10.26.5 textdomain

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [textdomain](#)(string *library*)

[textdomain\(\)](#) fixe le domaine à utiliser lors de recherche avec [gettext\(\)](#). Ce domaine dépend généralement de l'application. Le domaine par défaut précédent est retourné. Appelez cette fonction sans paramètre pour avoir la valeur courante, sans la modifier.

10.27 Fonctions HTTP

[\[Notes en ligne\]](#)

Ces fonctions permettent de travailler sur les informations transmises au navigateur, via le protocole HTTP.

10.27.1 header

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [header](#)(string *string*)

La fonction [header\(\)](#) permet de spécifier un entête HTTP lors de l'envoi des fichiers HTML. Reportez vous à [HTTP 1.1 Specification](#) pour plus d'informations sur les entêtes HTTP. NB : la fonction [header\(\)](#) doit être appelée avant la première balise HTML, et avant n'importe quel envoi de commande PHP. C'est une erreur très courante que de lire du code avec la fonction [include\(\)](#) ou avec `auto_prepend` et d'avoir des espace ou des lignes vides dans ce code qui produisent un début de sortie avant que [header\(\)](#) n'ai été appelé.

Il y a cependant deux entêtes spéciaux. Le premier est "Location". Non seulement il renvoie un entête au client, mais en plus, il envoie un statut de redirectoin à Apache. Du point de vue de l'auteur de script, cela importe peu, mais pour ceux qui connaissent les rouages internes d'Apache, c'est primordial.

```
header("Location: http://www.php.net"); /* Redirige le client vers le site PHP */
exit; /* Assure que le code ci dessous n'est jamais exécuté. */
```

Le deuxième type d'appel spécial regroupe tous les entêtes qui commencent par "HTTP/" (la casse n'est pas importante). Par exemple, si vous avez votre page d'erreur 404 Apache qui pointent sur un script PHP, c'est une bonne idée que de vous assurez que le script PHP génère une erreur 404. La première chose à dans votre script est :

```
header("http/1.0 404 Not Found");
```

Les scripts PHP génèrent souvent du HTML dynamiquement, qui ne doit pas être mis en cache, ni par le client, ni par les proxy intermédiaires. On peut forcer la désactivation du cache de nombreux clients et proxy avec

```
header("Expires: Mon, 26 Jul 1997 05:00:00 GMT"); // Date du passé
header("Last-Modified: " . gmdate("D, d M Y H:i:s") . " GMT"); // toujours modifié
header("Cache-Control: no-cache, must-revalidate"); // HTTP/1.1
```

```
header("Pragma: no-cache");
```

```
// HTTP/1.0
```

10.27.2 setcookie

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [setcookie\(\)](#) (string *name*, string *value*, int *expire*, string *path*, string *domain*, int *secure*)

[setcookie\(\)](#) définit un cookie qui sera envoyé avec le reste des entêtes. Les cookies doivent passer avant tout autre entête (c'est une restriction des cookies, pas de PHP). Cela vous impose d'appeler cette fonction avant toute balise <html> ou <head>.

Tous les arguments sauf *name* (nom) sont optionnels. Si seul le nom est présent, le cookie portant ce nom sera supprimé du client. Vous pouvez aussi utiliser une chaîne vide comme valeur, pour ignorer un argument. Le paramètre *expire* est un timestamp UNIX, du même genre que celui retourné par [time\(\)](#) ou [mktime\(\)](#). Le paramètre *secure* indique que le cookie doit être uniquement transmis à travers une connexion HTTPS sécurisée.

Erreurs communes:

Les cookies ne seront accessibles qu'au chargement de la prochaine page.

Les appels multiples à [setcookie\(\)](#) dans la même page seront réalisés dans l'ordre inverse. Si vous essayez d'effacer un cookie avant d'insérer une autre valeur, il faut placer l'insertion avant l'effacement.

Quelques exemples : *Exemples* [setcookie\(\)](#)

```
setcookie("TestCookie","Test Value");
setcookie("TestCookie",$value,time()+3600); /* expire in 1 hour */
setcookie("TestCookie",$value,time()+3600,"/~rasmus/",".utoronto.ca",1);
```

Notez que la partie "valeur" du cookie sera automatiquement encodée URL lorsque vous envoyez le cookie, et lorsque vous le recevez, il sera automatiquement décodé, et affecté à la variable du même nom que le cookie. Pour voir le résultat, essayez les scripts suivants :

```
echo $TestCookie;
echo $_HTTP_COOKIE_VARS["TestCookie"];
```

Vous pouvez aussi utiliser les cookies avec des tableaux, en utilisant la notation des tableaux. Cela a pour effet de créer autant de cookie que votre tableau a d'éléments, mais lorsque les cookies seront reçus par PHP, les valeurs seront placées dans un tableau :

```
setcookie( "cookie[three]", "cookiethree" );
setcookie( "cookie[two]", "cookietwo" );
setcookie( "cookie[one]", "cookieone" );
if ( isset( $cookie ) ) {
while( list( $name, $value ) = each( $cookie ) ) {
echo "$name == $value<br>\n";
}
}
```

Pour d'autres informations sur les cookies, jetez un oeil sur

http://www.netscape.com/newsref/std/cookie_spec.html.

Microsoft Internet Explorer 4 utilisé avec le Service Pack 1 ne gère pas bien les cookies qui possèdent un paramètre *path*.

Netscape Communicator 4.05 et Microsoft Internet Explorer 3.x semblent ne pas gérer correctement les cookies lorsque *path* et *time* ne sont pas fournis.

10.28 Fonctions Hyperwave

[\[Notes en ligne\]](#)

10.28.1 Introduction

[\[Notes en ligne\]](#)

Hyperwave a été développé par [ICM](#) à Graz. Son nom original était *Hyper-G* et il a pris le nom de Hyperwave lors de sa commercialisation (en 1996, si mes souvenirs sont bons).

Hyperwave n'est pas gratuit. La version actuelle est la 4.1, disponible à www.hyperwave.com. Une version limitée à 30 jours peut être demandée.

HIS est un système d'information similaire à une base de données, (HIS, Hyperwave Information Server).

HIS se concentre sur l'enregistrement et la gestion des documents. Un document peut être n'importe quelle donnée, qui peut être stockée dans un fichier. Chaque document est accompagné par un enregistrement. Cet enregistrement contient des méta données à propos du document. Ces méta données sont des listes d'attributs qui peuvent être étendues par l'utilisateur. Un attribut est une paire clé/valeur de la forme : clé =valeur.

L'enregistrement complet contient autant de paire que le désire l'utilisateur. Le nom d'un attribut n'a pas besoin d'être unique, c'est à dire qu'une même clé peut apparaître plusieurs fois dans un enregistrement. Cela peut être utile si vous devez donner un titre à votre document en plusieurs langues, par exemple. Dans un cas pareil, la convention est que chaque valeur de titre est précédée par deux lettres et deux points, tel que : 'fr:Titre en français' ou 'ge:Titel in deutsch'. D'autres attributs comme une description ou des mots clés sont aussi susceptibles de recourir à ce genre de procédé. Vous pouvez aussi remplacer l'abréviation de langage par une autre chaîne, tant qu'elle est séparée de la valeur par les deux points.

Chaque enregistrement a une représentation native qui contient toutes les paires clé/valeur, séparées par un retour à la ligne. L'extension Hyperwave reconnaît une autre représentation qui est un tableau associatif, où les attributs servent de clés. Les attributs multilingues étant géré sous la forme d'un autre tableau associatif, dont les clés sont les chaînes de langue. En fait, tous les attributs multiformes sont gérés sous la forme de tableau associatif. (Cela n'est pas encore complètement codé. Uniquement les attributs de titre, description et mot clés sont traités correctement).

En dehors des documents, tous les hyper liens contenus dans un documents sont enregistrés dans un autre enregistrement. Les hyperliens qui sont à l'intérieur d'un document en seront supprimés, et enregistrés dans des objets particuliers, au moment de l'insertion dans la base de données. L'enregistrement des hyper-liens contient les informations d'origine et d'objectif. Afin d'accéder au document original, vous devez lire le document sans les liens, puis lire les liens, et les réinsérer (les [hw_pipedocument\(\)](#) et [hw_gettext\(\)](#) le font pour vous. L'avantage de séparer les liens du document est évident : une fois qu'un document, cible d'un hyperlien, a été renommé, le liens peut facilement être modifié. Le document contenant le lien n'est pas modifié pour autant. Vous pouvez même ajouter un lien à un document sans le modifier.

Dire que [hw_pipedocument\(\)](#) et [hw_gettext\(\)](#) font l'insertion automatiquement n'est pas aussi simple qu'il n'y paraît. L'insertion implique une certaine hiérarchie de document. Sur un serveur web, la hiérarchie est fournie par le système de fichiers, mais Hyperwave dispose de sa propre hiérarchie et les noms de fichiers ne reflètent pas la position d'un objet dans cette hiérarchie. Ainsi, la création de liens requière en premier lieu la construction de la hiérarchie et de l'espace des noms dans une hiérarchie web et un espace de nom web. La différence fondamentale entre Hyperwave et le web est qu'il y a une distinction claire en les noms et la

hiérarchie dans Hyperwave. Le nom ne contient aucune information à propos de la position de l'objet dans la hiérarchie. Sur le web, le nom contient les informations de localisation dans la hiérarchie. Cela conduit à deux méthodes de d'accès : soit la hiérarchie Hyperwave et le nom de l'objet sont inscrit dans l'URL. Pour simplifier les choses, une deuxième approche est pratiquée. L'objet Hyperwave de nom 'mon_objet' correspond à l'URL 'http://hote/mon_objet', peu importe alors où il est rangé dans la hiérarchie. Un objet dont le nom est 'parent/mon_objet' peut être le fils de l'objet 'mon_objet' dans la hiérarchie Hyperwave, bien que ce soit le contraire en convention web, et cela risque de perturber l'utilisateur.

Ayant pris cette décision, un deuxième problème surgit : comment faire l'interface avec PHP ? L'URL `http://hote/mon_objet` n'appellera aucun script PHP à moins que vous ne demandiez à votre serveur web de le remplacer par autre chose, comme par exemple : `'http://host/php3_script/mon_objet'` et le script `'php3_script'` utilise la variable `$PATH_INFO` pour rechercher l'objet 'mon_objet' sur le serveur Hyperwave. Il y a juste un petit inconvénient, qui peut facilement être corrigé. Réécrire un URL ne vous permettra aucun accès aux autres documents du serveur web. Un script de recherche dans le serveur Hyperwave serait impossible. Il vous faudra donc au moins une autre règle pour exclure certaines URL, comme par exemple celles qui commencent par `http://host/Hyperwave`. Voici de manière simple, la manière de partager un espace de nom entre un serveur web et un serveur Hyperwave serveur.

Basé sur le mécanisme précédent, on trouve l'insertion dans les documents.

Il est plus compliqué d'avoir PHP ne fonctionne pas comme un module de serveur, ou un scrip CGI, mais comme une application indépendante. Dans ce cas, il est utile d'inscrire la hiérarchie et le nom de fichier Hyperwave dans le système de fichier. Mais comme cela risque d'entrer en conflit avec le séparateur de dossier ('/'), il faut le remplacer par un autre caractère, '_ '.

Le protocole réseau pour communiquer avec un serveur Hyperwave est appelé [HG-CSP](#) (Hyper-G Client/Server Protocol). Il est basé sur des messages qui initie des actions, comme par exemple, lire l'en-tête de fichier. Dans les premières versions de Hyperwave Server deux clients natifs (Harmony, Amadeus) étaient fournis pour permettre la communication avec le serveur. Ils ont disparu lors de la commercialisation de Hyperwave. En tant qu'ersatz, un client appelé wavemaster est désormais fourni. wavemaster est un espèce ce convertisseur de protocole de HTTP en HG-CSP. L'idée est de faire toute l'administration de la base et la visualisation des documents par une interface web. Le wavemaster implémente un jeu d'emplacement pour certaines actions de personnalisation de l'interface. Ce jeu est appelé PLACE language. PLACE pêche encore par le manque de nombreuses fonctions de programmations, et le manque d'évolutivité. Cela a conduit à l'utilisation de JavaScript ce qui ne rend pas la vie facile.

Que PHP supporte Hyperwave permet de combler ces manques. PHP implémente tous les messages définis par HG-CSP mais fourni d'autres commandes puissantes, comme par exemple, celle de lire des documents complets.

Hyperwave dispose de sa propre terminologie pour localiser certaines informations. Cette terminologie a été largement étendue. Presque toutes les fonctions utilisent l'un des types de données suivants :

- object ID: un entier, unique pour chaque objet sur le serveur Hyperwave. C'est aussi un des attributs de l'enregistrement de l'objet (ObjectID). Les object ids sont souvent utilisées comme paramètre d'entrée pour spécifier un objet.
- object record: Une chaîne contenant des paires clé=valeur. Les paires sont séparées par un retour à la ligne. Un enregistrement d'objet peut facilement être converti en tableau, avec la fonction [hw_objrec2array\(\)](#). De nombreuses fonctions retournent un objet records. Ces fonctions ont leur nom qui finit par obj.
- object array: Un tableau associatif qui contient tous les attributs d'un objet. La clé est l'attribut. Si un attribut apparaît plusieurs fois, il sera représenté sous la forme d'un tableau associatif ou indexé. Les attributs qui dépendent des langues (comme title, keyword ou description) seront représentés sous la forme d'un tableau associatif, dont les clés seront les abréviations de langues. Tous les autres attributs à valeur multiple prendront la forme d'un tableau indexé.
- hw_document: Ce type est un nouveau type de données, qui contient le document lui même, comme par exemple HTML, PDF etc. Il est optimisé pour les documents HTML mais peut s'utiliser avec

n'importe quel format.

De nombreuses fonctions qui retournent un tableau d'enregistrements, retournent aussi un tableau associé, avec des informations statistiques. Ce tableau est le dernier élément du tableau d'enregistrements. Les statistiques contiennent les entrées suivantes :

Hidden

- Nombre d'objet dont l'attribut PresentationHints est Hidden.
CollectionHead
- Nombre d'objet dont l'attribut PresentationHints est CollectionHead.
FullCollectionHead
- Nombre d'objet dont l'attribut PresentationHints est FullCollectionHead.
CollectionHeadNr
- Index du premier objet du tableau d'enregistrement avec l'attribut PresentationHints à
CollectionHead.
FullCollectionHeadNr
- Index du premier objet du tableau d'enregistrement avec l'attribut PresentationHints est
FullCollectionHead.
Total
- Total: Nombre d'enregistrements.

[10.28.2 Intégration avec Apache](#)

[\[Notes en ligne\]](#)

L'extension Hyperwave est utilisée de manière optimale lorsque PHP est compilé comme module Apache. Dans ce cas, le serveur Hyperwave sous jacent peut être caché quasiment totalement aux utilisateurs, si Apache utilise son moteur d'écriture. Les explications suivantes vous éclaireront :
Etant donné que PHP avec l'extension Hyperwave et Apache tend à remplacer la solution native basé sur Wavemaster, je vais supposer que le serveur Apache servira seulement d'interface Hyperwave. Ce n'est pas nécessaire, mais cela simplifie grandement la configuration. Le concept est très simple. Premièrement, vous avez besoin d'un script PHP qui évalue la variable *PATH_INFO* et considère que cette valeur est un objet Hyperwave. Appelons ce script 'Hyperwave'. L'URL `http://votre.hote/Hyperwave/nom_objet` retourne alors l'objet Hyperwave dont le nom est 'nom_objet'. Le script doit alors réagir suivant le type de l'objet. Si c'est un groupe, il devra probablement retourner une liste de fils. Si c'est un document, il pourra retourner son type MIME et son contenu. Une amélioration peut être obtenue en utilisant le moteur de réécriture d' Apache. D'un point de vue utilisateur, il est plus direct si l'URL `http://votre.hote/nom_objet` retourne l'objet. La règle de réécriture est simple :

```
RewriteRule ^/(.*) /usr/local/apache/htdocs/HyperWave/$1 [L]
```

Maintenant toutes les URL pointent sur un objet Hyperwave. Cela conduit à un problème simple. Il n'y a pas d'autre façon d'exécuter, c'est à dire rechercher, un autre script que ce script 'Hyperwave'. Cela pourra être corrigé avec une autre règle telle que:

```
RewriteRule ^/hw/(.*) /usr/local/apache/htdocs/hw/$1 [L]
```

Le dossier ``/usr/local/apache/htdocs/hw'` sera ainsi réservé pour d'autres scripts et fichiers. Assurez vous que cette règle est évaluée avant la première règle que nous avons définie. Il y a juste un léger inconvénient : tous les objets Hyperwave qui commencent par 'hw/' seront cachés. Alors, assurez vous que vous n'utilisez pas de tels noms. Si vous avez besoin d'autres dossiers, par exemple, un dossier d'images, ajoutez simplement d'autres règles. N'oubliez pas de lancer le moteur de réécriture avec

```
RewriteEngine on
```

Mon expérience m'a montré que vous aurez besoin des scripts suivants :

- Retourne l'objet lui même
- Pour autoriser la recherche
- S'identifier
- Choisir une configuration
- Un script pour chaque fonction supplémentaire, comme afficher un objet, afficher des informations sur les utilisateurs, afficher le statut du serveur, etc÷

[10.28.3 A faire](#)

[\[Notes en ligne\]](#)

Il reste encore beaucoup à faire :

- La fonction `hw_InsertDocument` doit être séparée en deux : [hw_insertobject\(\)](#) et `@xref{function.hw-putdocument , , hw_putdocument()}`.
- Les noms de nombreuses fonctions ne sont pas encore fixés.
- La plupart des fonctions requièrent la connexion courante comme premier paramètre. Cela conduit à beaucoup de frappe clavier, même si il n'y a souvent qu'une seule connexion en jeu. Une connexion par défaut améliorerait ceci.

[10.28.4 hw Array2Objrec](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

strin [hw_array2objrec](#) (array *object_array*)

Converti un *object_array* en un objet. Les attributs multiples tels que 'Title' en différentes langues seront traités correctement.

Voir aussi [hw_objrec2array\(\)](#).

[10.28.5 hw Children](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_children](#) (int *connection*, int *objectID*)

Retourne un tableau avec des object ids. Chaque object id est celui d'un des fils du groupe dont l'id est *objectID*. Ce tableau contient tous les fils, documents et groupes.

[10.28.6 hw_ChildrenObj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_childrenobj](#)(int *connection*, int *objectID*)

Retourne un tableau avec des object records. Chaque object records est celui d'un des fils du groupe dont l'id est *objectID*. Ce tableau contient tous les fils, documents et groupes

[10.28.7 hw_Close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_close](#)(int *connection*)

Retourne false si la connexion n'est pas valide, et sinon, true. Ferme la connexion *connection* à un serveur Hyperwave.

[10.28.8 hw_Connect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_connect](#)(string *host*, int *port*, string *username*, string *password*)

Ouvre une connexion Hyperwave et retourne un identifiant de connexion, en cas de succès, ou false, si la connexion n'a pas pu être créée. Chaque argument doit être entouré de guillemets, sauf le numéro de port. Les arguments *username* et *password* sont optionnels, et peuvent être ignorés. Dans ce cas, aucune identification ne sera faite au niveau du serveur. Cela revient à s'identifier en tant qu'utilisateur anonyme. Cette fonction retourne un identifiant de connexion qui sera nécessaire aux autres fonctions Hyperwave. Vous pouvez avoir plusieurs connexions simultanées. N'oubliez pas que les mots de passe ne sont pas cryptés.

Voir aussi [hw_pconnect\(\)](#).

[10.28.9 hw_Cp](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_cp](#)(int *connection*, array *object_id_array*, int *destination id*)

Copie les objet ayant les objects id *object_id_array*, et crée un groupe ayant l'object id *destination id*. La valeur retournée est le nombre d'objets copiés.

Voir aussi [hw_mv\(\)](#).

[10.28.10 hw_Deleteobject](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_deleteobject](#)(int *connection*, int *object_to_delete*)

Efface l'objet dont l'identifiant est *object_to_delete*. Toutes les instances de l'objets seront effacées. Retourne TRUE si aucune erreur ne survient, et sinon, FALSE.

Voir aussi [hw_mv\(\)](#).

[10.28.11 hw DocByAnchor](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_docbyanchor](#) (int *connection*, int *anchorID*)

Retourne l'identifiant d'objet de l'objet dans l'ancrage *anchorID*.

[10.28.12 hw DocByAnchorObj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_docbyanchorobj](#) (int *connection*, int *anchorID*)

Retourne les attributs du document qui correspond à *anchorID*.

[10.28.13 hw DocumentAttributes](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_documentattributes](#) (int *hw_document*)

Retourne les attributs du document.

Voir aussi [hw_documentbodytag\(\)](#), [hw_documentsize\(\)](#).

[10.28.14 hw DocumentBodyTag](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_documentbodytag](#) (int *hw_document*)

Retourne la balise BODY du document. Si le document est un document HTML , la balise BODY doit être affichée avant le document.

Voir aussi [hw_documentattributes\(\)](#), [hw_documentsize\(\)](#).

[10.28.15 hw DocumentContent](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_documentcontent](#) (int *hw_document*)

Retourne la balise BODY du document. Si le document est un document HTML , la balise BODY doit être affichée avant le document.

Voir aussi [hw_documentattributes\(\)](#), [hw_documentsize\(\)](#), [hw_documentsetcontent\(\)](#).

[10.28.16 hw DocumentSetContent](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_documentsetcontent](#) (int *hw_document*, string *content*)

Modifie/remplace le contenu d'un document. Si le document est un document HTML, le contenu représente

tout qui est placé au delà de la balise BODY. Les informations de HEAD et de la balise BODY sont enregistrés dans les attributs. Si vous fournissez aussi ces informations dans le corps du document, le serveur Hyperwave modifiera les attributs. Cela n'est cependant pas une bonne idée. Si la fonction échoue, l'ancien contenu est restauré.

Voir aussi [hw_documentattributes\(\)](#), [hw_documentsize\(\)](#), [hw_documentcontent\(\)](#).

10.28.17 hw DocumentSize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_documentsize](#)(int *hw_document*)

Retourne la taille du document en octets.

Voir aussi [hw_documentbodytag\(\)](#), [hw_documentattributes\(\)](#).

10.28.18 hw ErrorMessage

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_errormsg](#)(int *connection*)

Retourne une chaîne contenant le dernier message d'erreur, ou 'No Error' (pas d'erreur). Si false est retourné, cette fonction a échoué. Ce message est relatif à la dernière commande exécutée.

10.28.19 hw EditText

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_edittest](#)(int *connection*, int *hw_document*)

Charge le texte du document sur le serveur. Les attributs du document ne doivent pas être modifiés tant que le document est en train d'être édité. Cette fonction n'est disponible que sur les documents texte. Elle n'ouvrira pas de canal de transfert, et donc, bloquera le script durant le transfert.

Voir aussi [hw_pipedocument\(\)](#), [hw_free_document\(\)](#), [hw_documentbodytag\(\)](#), [hw_documentsize\(\)](#), [hw_outputdocument\(\)](#), [hw_gettext\(\)](#).

10.28.20 hw Error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_error](#)(int *connection*)

Retourne le code d'erreur de la dernière erreur. Si la valeur 0 est retournée, c'est qu'il n'y avait pas d'erreur. L'erreur se rapporte à la dernière commande.

10.28.21 hw Free Document

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_free_document](#)(int *hw_document*)

Détruit un document Hyperwave.

[10.28.22 hw_GetParents](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getparentsobj](#)(int *connection*, int *objectID*)

Retourne un tableau indexé avec les identifiants des objets parents de *objectID*.

[10.28.23 hw_GetParentsObj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getparentsobj](#)(int *connection*, int *objectID*)

Retourne un tableau indexé, avec les attributs et un tableau associé, d'informations statistiques à propos des attributs. Ce tableau associé est le dernier élément du tableau retourné. Chaque attribut appartient au père de l'objet *objectID*.

[10.28.24 hw_GetChildColl](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getchildcoll](#)(int *connection*, int *objectID*)

Retourne un tableau contenant les identifiants d'objets des groupes fils du groupe *objectID*. Cette fonction ne retournera pas d'identifiants d'objets des documents fils.

Voir aussi [hw_getchilddoccoll\(\)](#).

[10.28.25 hw_GetChildCollObj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getchildcollobj](#)(int *connection*, int *objectID*)

Retourne un tableau d'object record. Chaque object records appartient à un groupe d'enfants de la collection *objectID*. La fonction ne retournera pas de documents enfants.

Voir aussi [hw_childrenobj\(\)](#), [hw_getchilddoccollobj\(\)](#).

[10.28.26 hw_GetRemote](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_getremote](#)(int *connection*, int *objectID*)

Retourne un document distant. Les documents distants sont, en Hyperwave, des documents lus depuis une source externe. La plus part des documents éloignés sont des pages web externes, ou des requêtes sur une base de données. Afin de pouvoir accéder à des sources externes, gr'ces aux documents distants, Hyperwave introduit l'interface HGI (Hyperwave Gateway Interface) qui est similaire à CGI. Actuellement, seuls les protocoles de ftp, http et certaines bases de données sont accessibles avec HGI. [hw_getremote\(\)](#) retourne le document de la source distante. Si vous voulez utiliser cette fonction, il vous faut vous familiariser HGIs. Il est aussi préférable d'utiliser PHP plutôt que Hyperwave pour accéder aux sources externes. Le support des bases de données sera plus difficile avec Hyperwave que PHP.

Voir aussi [hw_getremotechildren\(\)](#).

[10.28.27 hw_GetRemoteChildren](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_getremotechildren](#)(int *connection*, string *object record*)

Retourne les fils d'un document distant. Les fils d'un document distant sont des documents distants eux mêmes. Cela est cohérent si une requête sur une base de données doit être rendu plus sélective, comme expliqué dans Hyperwave Programmers' Guide. Si le nombre de fils est 1 la fonction va retourner le document lui même, la fonction retournera le document lui même, formaté Hyperwave Gateway Interface (HGI). Si le nombre de fils est supérieur 1 la fonction retournera un tableau d'attributs, qui pourra servir à une nouvelle requête avec [hw_getremotechildren\(\)](#). Ces attributs sont virtuels et n'existent pas sur le serveur Hyperwave, et ainsi, n'ont pas d'identifiant d'objet valide. L'ordre exact de ces objets est du ressort de HGI. Si vous voulez utiliser cette fonction, vous devez être très familier HGIs. Il vaut mieux PHP plutôt que Hyperwave pour accéder aux fichiers distants. Le support de base de données y est bien meilleur.

Voir aussi [hw_getremote\(\)](#).

[10.28.28 hw_GetSrcByDestObj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getsrcbydestobj](#)(int *connection*, int *objectID*)

Retourne les attributs de tous les ancrages qui pointent sur *objectID*. L'objet peut être un document ou un autre ancrage, de type destination.

Voir aussi [hw_getanchors\(\)](#).

[10.28.29 hw_GetObject](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getobject](#)(int *connection*, [int]array] *objectID*, string *query*)

Retourne les attributs de l'objet dont l'identifiant est *objectID*, si le second paramètre est un entier. Si le second paramètre est un tableau, la fonction retournera un tableau d'attributs. Dans ce cas, le dernier paramètre est aussi évalué.

query a la syntaxe suivante :

```
<expression> ::= "(" <expression> ")" |
"!<expression> | /* NOT */
<expression> "||" <expression> | /* OR */
<expression> "&&" <expression> | /* AND */
<attribute> <operator> <value>
<attribute> ::= /* * n'importe quel attribut (Title, Author, DocumentType ...) */
<operator> ::= "=" | /* égal */
"<" | /* moins que (comparaison de type chaîne) */
">" | /* plus que (comparaison de type chaîne) */
"~" /* recherche par expression régulière */
```

query permet de sélectionner une nouvelle fois certains objets dans la liste des objets donnés. Contrairement

aux autres requêtes, celle ci peut utiliser des attributs non indexés. Le nombre d'attributs retourné dépend de la requête de la requête, et des autorisations d'accès aux objets.

Voir aussi [hw_getandlock\(\)](#), [hw_getobjectbyquery\(\)](#).

10.28.30 hw_GetAndLock

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_getandlock](#) (int *connection*, int *objectID*)

Retourne les attributs, et verrouille l'objet *objectID*. Le verrouillage empêchera les autres utilisateurs d'y accéder, jusqu'à ce qu'il soit deverrouillé.

Voir aussi [hw_unlock\(\)](#), [hw_getobject\(\)](#).

10.28.31 hw_GetText

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_gettext](#) (int *connection*, int *objectID*, mixed *rootID/prefix*)

Retourne le document de l'objet *objectID*. Si le document possède des ancrages qui peuvent être insérés, ils seront déjà insérés. L'option *rootID/prefix* peut être une chaîne ou un entier. Si c'est un entier, il détermine la méthode d'insertion des liens dans le document. Par défaut, il vaut 0 et les liens seront construits en fonction du nom de l'objet cible. Cela sert beaucoup dans les applications web. Si un lien pointe sur un objet avec le nom 'film_internet' le lien HTML sera . La position réelle de la source et de la cible dans la hiérarchie seront ignorés. Vous devrez modifier votre site web pour qu'il réécrit les URL, comme par exemple '/mon_script.php3/film_internet'. 'mon_script.php3' devra analyser \$PATH_INFO et savoir recherche le document '/mon_script.php3/film_internet'. Si vous ne voulez pas de ce comportement, vous pouvez affecter à *rootID/prefix* n'importe quel préfixe. Dans ce cas, ce sera une chaîne.

Si *rootID/prefix* est un entier différent de 0 le lien sera construit avec tous les noms de la hiérarchie, en commençant à l'objet d'identifiant *rootID/prefix*, et séparé par des slash. Si, par exemple, le document 'film_internet' est situé à 'a-b-c-internet_movie' et '-' qui sert de séparateur hiérarchique de niveau sur le serveur Hyperwave et le document source est situé dans 'a-b-d-source' alors, le lien HTML serait: . Cela est très pratique si vous voulez télécharger tout le contenu d'un serveur sur un disque, et faire une carte du système sur votre disque.

Cette fonction n'est opérationnelle qu'avec des document de pur texte. Elle n'ouvrira pas de canal spécial de transfert, et ainsi, bloquera le script le temps du transfert.

Voir aussi [hw_pipedocument\(\)](#), [hw_free_document\(\)](#), [hw_documentbodytag\(\)](#), [hw_documentsize\(\)](#), [hw_outputdocument\(\)](#).

10.28.32 hw_GetObjectByQuery

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getobjectbyquery](#) (int *connection*, string *query*, int *max_hits*)

Recherche un objet sur tout le serveur et retourne un tableau d' object ids. Le nombre maximum d'objet est limité par *max_hits*. Si *max_hits* vaut -1 il n'y a pas de limite.

La requête ne fonctionnera qu'avec des attributs indexés.

Voir aussi [hw_getobjectbyqueryobj\(\)](#).

[10.28.33 hw_GetObjectByQueryObj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getobjectbyqueryobj](#) (int *connection*, string *query*, int *max_hits*)

Recherche un objet sur tout le serveur et retourne un tableau d' object records. Le nombre maximum d'objet est limité par *max_hits*. Si *max_hits* vaut -1 il n'y a pas de limite.

La requête ne fonctionnera qu'avec des attributs indexés.

Voir aussi [hw_getobjectbyquery\(\)](#).

[10.28.34 hw_GetObjectByQueryColl](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getobjectbyquerycoll](#) (int *connection*, int *objectID*, string *query*, int *max_hits*)

Recherche un objet sur tout le groupe objectID et retourne un tableau d' object records. Le nombre maximum d'objet est limité par *objectID*. Si *objectID* vaut -1 il n'y a pas de limite.

La requête ne fonctionnera qu'avec des attributs indexés.

Voir aussi [hw_getobjectbyquerycollobj\(\)](#).

[10.28.35 hw_GetObjectByQueryCollObj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getobjectbyquerycollobj](#) (int *connection*, int *objectID*, string *query*, int *max_hits*)

Recherche un objet sur tout le groupe objectID et retourne un tableau d' object records. Le nombre maximum d'objet est limité par *objectID*. Si *objectID* vaut -1 il n'y a pas de limite.

La requête ne fonctionnera qu'avec des attributs indexés.

Voir aussi [hw_getobjectbyquerycoll\(\)](#).

[10.28.36 hw_GetChildDocColl](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getchilddoccoll](#) (int *connection*, int *objectID*)

Retourne un tableau avec les id des documents fils d'une collection.

Voir aussi [hw_getchildcoll\(\)](#).

[10.28.37 hw_GetChildDocCollObj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getchilddoccollobj](#) (int *connection*, int *objectID*)

Retourne un tableau contenant les attributs des documents fils du groupe *objectID*.

Voir aussi [hw_childrenobj\(\)](#), [hw_getchildcollobj\(\)](#).

[10.28.38 hw_GetAnchors](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getanchors](#) (int *connection*, int *objectID*)

Retourne un tableau contenant les identifiants des ancrages du document *objectID*.

[10.28.39 hw_GetAnchorsObj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getanchorsobj](#) (int *connection*, int *objectID*)

Retourne un tableau d'attributs des ancrages du document *objectID*.

[10.28.40 hw_Mv](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_mv](#) (int *connection*, array *object id array*, int *source id*, int *destination id*)

Déplace les objets dont les identifiants sont passés dans le tableau *source id*, depuis le *source id* dans le *destination id*. Si destination id vaut 0, les objets ne seront plus insérés dans le groupe (ni dans le serveur). Dans ce cas, si une instance était la dernière instance d'un objet, l'objet sera effacé. Si vous voulez effacer toutes les instances d'un coup, utilisez [hw_deleteobject\(\)](#).

La valeur retournée est le nombre d'objet déplacés.

Voir aussi [hw_cp\(\)](#), [hw_deleteobject\(\)](#).

[10.28.41 hw_Identify](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_identify](#) (string *username*, string *password*)

Identifie un utilisateur, dont le nom d'utilisateur est *username* and et le mot de passe *password*.

L'identification n'est valide que pour la session en cours. Je ne pense pas que cette fonction serve souvent.

Dans la plus part des cas, il est plus simple de s'identifier lors de l'ouverture de la connexion.

Voir aussi [hw_connect\(\)](#).

[10.28.42 hw_InCollections](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_incollections](#) (int *connection*, array *object_id_array*, array *collection_id_array*, int *return_collections*)

Vérifie qu'un ensemble d'objet (documents ou groupes) donnés par *object_id_array* fait partie des groupe listés par *object_id_array*. Lorsque le quatrième paramètre *return_collections* vaut 0, le sous ensemble d'identifiant qui font partie d'un groupe (i.e. les documents ou groupes qui sont fils d'un ou plusieurs groupe,

ou leurs fils, récursivement) est retourné sous la forme d'un tableau. Cette option permet de mettre en valeur la partie de l'arborescence qui contient le résultat d'une requête, dans un sens graphique.

[10.28.43 hw_Info](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_info](#) (int *connection*)

Retourne les informations de la connexion courante. La chaîne retournée a le format suivant : <Serverstring>, <Host>, <Port>, <Username>, <Port of Client>, <Byte swapping>

[10.28.44 hw_InsColl](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_inscoll](#) (int *connection*, int *objectID*, array *object_array*)

Insère un nouveau groupe, avec les attributs *object_array* dans le groupe *objectID*.

[10.28.45 hw_InsDoc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_insdoc](#) (int *connection*, int *parentID*, string *object_record*, string *text*)

Insère un nouveau document avec les attributs *object_record*, dans le groupe *parentID*. Cette fonction insère soit un objet avec ses seuls attributs, soit pur objet ascii, avec *text* si il est fourni. Si vous voulez insérer un document de type général, utilisez plutôt [hw_insertdocument\(\)](#).

Voir aussi [hw_insertdocument\(\)](#), [hw_inscoll\(\)](#).

[10.28.46 hw_InsertDocument](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_insertdocument](#) (int *connection*, int *parent_id*, int *hw_document*)

Insère un document dans le groupe *parent_id*. Le document doit avoir été créé auparavant avec [hw_new_document\(\)](#). Assurez vous que les attributs du nouveau document contiennent au moins les attributs suivants : Type, DocumentType, Title et Name. Vous aurez aussi parfois besoin de MimeType. La fonction retourne l'identifiant de l'objet inséré, ou bien false.

Voir aussi [hw_pipedocument\(\)](#).

[10.28.47 hw_InsertObject](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_insertobject](#) (int *connection*, string *object rec*, string *parameter*)

Insère un objet dans le serveur. L'objet peut être n'importe quel objet Hyperwave valide. Reportez vous à la documentation HG-CSP pour plus de détails sur les paramètres.

Note: Si vous voulez insérer un ancre, l'attribut Position doit être mis à la valeur start/end (début ou fin) ou encore 'invisible'. Les positions invisibles sont nécessaire si l'annotation n'a pas de liens correspondant dans le texte de l'annotation.

Voir aussi [hw_pipedocument\(\)](#), [hw_insertdocument\(\)](#), [hw_insdock\(\)](#), [hw_inscoll\(\)](#).

10.28.48 hw_mapid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_mapid](#)(int *connection*, int *server id*, int *object id*)

Représente l'id d'un objet global de n'importe quel serveur Hyperwave, même si vous ne vous y êtes pas connecté avec [hw_connect\(\)](#), avec un id d'objet local virtuel. Cet id d'objet local peut alors être utilisé comme n'importe quel id d'objet : par exemple on peut obtenir l'enregistrement d'objet avec la fonction [hw_getobject\(\)](#). L'id du serveur est la première partie de l'id global (GOid) de l'objet, qui est en fait une adresse IP.

Note: Afin d'utiliser cette fonction, vous devez lever le flag F_DISTRIBUTED, ce qui ne peut être fait qu'à la compilation. Par défaut, il n'est pas levé. Lisez les commentaires dans le fichier hg_comm.c

10.28.49 hw_Modifyobject

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_modifyobject](#)(int *connection*, int *object_to_change*, array *remove*, array *add*, int *mode*)

Cette commande permet d'effacer, d'ajouter ou de modifier des attributs d'un objet. L'objet est repéré par son identifiant *object_to_change*. Le premier tableau, *remove*, est la liste des attributs à effacer. Le deuxième tableau *add* est la liste des attributs à ajouter. Afin de modifier un attribut, il vous faudra d'abord l'effacer, puis l'ajouter à nouveau. [hw_modifyobject\(\)](#) effacera toujours les attributs avant de les ajouter, à moins que la valeur de l'attribut à effacer ne soit pas une chaîne, ou un tableau.

Le dernier paramètre détermine si la modification est récursive ou pas. 1 signifie que la modification est récursive. Si un objet ne peut pas être modifié, il sera ignoré. [hw_error\(\)](#) n'indiquera alors pas toujours d'erreur, même si certains objets n'ont pas pu être modifiés.

Les clés des deux tableaux sont les noms des attributs. Les valeurs de chaque élément peuvent être un tableau, ou une chaîne ou n'importe quoi d'autre. Dans le cas du tableau, la valeur de l'attribut est construite en séparant chaque élément par un point virgule. Dans le cas de la chaîne, elle sert directement de valeur. Une chaîne vide provoquera un effacement de l'attribut. Dans le cas où la valeur n'est ni un tableau, ni une chaîne, aucune opération ne sera effectuée. Cela est nécessaire si vous voulez ajouter un attribut complètement une nouvelle valeur pour un attribut existant. Si le tableau d'effacement contenait une chaîne vide comme attribut, le serveur tenterait d'effacer l'attribut, ce qui échouerait de toute manière, car cet attribut n'existe pas. L'ajout de cet attribut échouerait aussi. Affecter la valeur de 0 à cet attribut ne l'effacerait pas, et l'ajout fonctionnerait. Si vous voulez changer l'attribut 'Nom' de valeur courante 'livres' en 'articles' vous devrez faire deux tableaux, et appeler [hw_modifyobject\(\)](#). *Modification d'un attribut*

```
// $connect est une connexion valide
// $objid est l'identifiant de l'objet
$remarr = array("Name" => "books");
$addarr = array("Name" => "articles");
$hw_modifyobject($connect, $objid, $remarr, $addarr);
```

Afin d'effacer/ajouter une paire nom=valeur aux attributs d'un objet, utilisez simplement les tableaux d'effacement et d'ajout, et laissez le dernier/troisième paramètre vide. Si l'attribut est le premier de ce nom à ajouter, donnez une valeur entière à cet élément. **Ajouter un nouvel attribut**

```
// $connect is an existing connection to the Hyperwave server
// $objid is the ID of the object to modify
$remarr = array("Name" => 0);
$addarr = array("Name" => "articles");
$hw_modifyobject($connect, $objid, $remarr, $addarr);
```

Note : *Les attributs multilingues, (tels que 'Title'), peuvent être modifiés de deux manières : soit en fournissant la valeur de ces attributs de manière native (langue :valeur), soit en fournissant un tableau avec les éléments de chaque langue, comme décrit ci-dessus. L'exemple deviendra alors :*
modifying Title attribute

```
$remarr = array("Title" => "en:Books");
$addarr = array("Title" => "en:Articles");
$hw_modifyobject($connect, $objid, $remarr, $addarr);
```

ou **Modifier l'attribut Title**

```
$remarr = array("Title" => array("en" => "Books"));
$addarr = array("Title" => array("en" => "Articles", "ge"=>"Artikel"));
$hw_modifyobject($connect, $objid, $remarr, $addarr);
```

Pour supprimer l'entrée française 'Livres' et ajouter l'entrée 'Articles' et l'entrée allemande 'Artikel'.
Suppression d'un attribut

```
$remarr = array("Title" => "");
$addarr = array("Title" => "en:Articles");
$hw_modifyobject($connect, $objid, $remarr, $addarr);
```

Note : *Cet exemple va effacer tous les attributs avec le nom 'Title' et ajouter un nouvel attribut 'Title'. Cela peut être pratique pour effacer des attributs récursivement.*

Note : *Si vous devez effacer tous les attributs avec un certains nom, vous devez passer une chaîne vide comme valeur.*

Note : *Seul les attributs 'Title', 'Description' et 'Keyword' gère correctement le préfixe de langue. Pour les autres attributs qui ne portent pas de préfixe de langage, le préfixe 'xx' sera assigné.*

Note : *L'attribut 'Name' est un peu particulier. Dans certains cas, il ne peut pas être complètement effacé. Vous aurez alors le message 'Change of base attribute' (l'apparition de cette erreur n'est pas très claire). Ainsi, vous aurez à ajouter une nouvelle entrée pour Name puis, effacer l'ancien.*

Note : *Il ne faut pas encadrer cette fonction par des appels à [hw_getandlock\(\)](#) et [hw_unlock\(\)](#). [hw_modifyobject\(\)](#) le fait de manière interne.*

Retourne TRUE si aucune erreur ne survient, et FALSE sinon.

[10.28.50 hw New Document](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_new_document](#) (string *object_record*, string *document_data*, int *document_size*)

Retourne un nouveau document Hyperwave avec comme données *document_data* et comme attributs *object_record*. La longueur de *document_data* doit être donnée dans *document_size*. Cette fonction n'insère pas l'objet dans le serveur Hyperwave.

Voir aussi [hw_free_document\(\)](#), [hw_documentsize\(\)](#), [hw_documentbodytag\(\)](#), [hw_outputdocument\(\)](#), [hw_insertdocument\(\)](#).

[10.28.51 hw Objrec2Array](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_objrec2array](#) (string *object_record*)

Convertir les attributs *object_record* d'un objet en un tableau. Les clés du tableau seront les noms des attributs. Les attributs multiples comme par exemple 'Title', dans différentes langues, seront rassemblées dans un autre tableau. Une clé est la partie gauche d'un attribut. Actuellement, seuls les attributs 'Title', 'Description' et 'Keyword' sont traités correctement.

Voir aussi [hw_array2objrec\(\)](#).

[10.28.52 hw OutputDocument](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_outputdocument](#) (int *hw_document*)

Affiche *hw_document* sans la balise BODY.

[10.28.53 hw pConnect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_pconnect](#) (string *host*, int *port*, string *username*, string *password*)

Retourne un index de connexion en cas de succès, et false si la connexion n'a pas pu être créée. Ouvre une connexion persistante à un serveur Hyperwave. Tous les arguments doivent être entre guillemets, hormis le numéro de port (*port*). Les arguments *username* et *password* sont optionnels, et peuvent être ignorés. Dans ce cas, aucune authentification ne sera faite, (connexion anonyme). Cette fonction retourne un index de connexion qui sera utilisée par les autres fonctions Hyperwave. Vous pouvez ouvrir plusieurs connexions simultanées.

Voir aussi [hw_connect\(\)](#).

[10.28.54 hw PipeDocument](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_pipedocument](#) (int *connection*, int *objectID*)

Retourne le document Hyperwave d'objet id *objectID*. Si le document a des ancres, ils seront insérés. Le document sera transmis via une connexion de données spéciale, qui ne bloque pas la connexion de contrôle (ie, le serveur n'attend pas la fin du transfert pour rendre la main).

Voir aussi [hw_gettext\(\)](#) (insertions), [hw_free_document\(\)](#), [hw_documentsize\(\)](#), [hw_documentbodytag\(\)](#), [hw_outputdocument\(\)](#).

10.28.55 hw Root

[\[Notes en ligne\]](#) [\[Exemples\]](#)

```
int hw\_root ()
```

Retour l' Object id de la racine. Actuellement, cette identifiant est toujours 0. L'ensemble des fils de la racine est celui du serveur courant.

10.28.56 hw Unlock

[\[Notes en ligne\]](#) [\[Exemples\]](#)

```
int hw\_unlock (int connection, int objectID)
```

Déverrouille un document, et laisse l'accès aux autres utilisateurs.

Voir aussi [hw_getandlock\(\)](#).

10.28.57 hw Who

[\[Notes en ligne\]](#) [\[Exemples\]](#)

```
int hw\_who (int connection)
```

Retourne un tableau contenant la liste des utilisateurs actuellement connectés au serveur Hyperwave. Chaque élément du tableau est lui même un tableau, qui contient l'identifiant de l'élément, le nom, le système, la date de connexion (onSinceDate), l'heure de connexion (onSinceTime), la durée de connexion (TotalTime) et self. 'self' contient 1 l'élément est l'utilisateur qui a appelé la fonction.

10.28.58 hw Username

[\[Notes en ligne\]](#) [\[Exemples\]](#)

```
string hw\_getusername (int connection)
```

Retourne le nom d'utilisateur utilisé par la connexion.

10.29 Fonctions InterBase

[\[Notes en ligne\]](#)

Interbase est une base de données populaire, créée par Borland/Inprise. Pour plus d'informations sur Interbase, allez à <http://www.interbase.com>. Par ailleurs, Interbase vient de rejoindre le mouvement Open Source!

[10.29.1 ibase_connect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_connect](#) (string *database*, string *username* , string *password*)

Ouvre une connexion à une base de données Interbase. *Exemple avec [ibase_connect\(\)](#)*

```
$dbh = ibase_connect ($host, $username, $password);
$stmt = 'SELECT * FROM tblname';
$stmt = ibase_query ($dbh, $stmt);
while ($row = ibase_fetch_object ($sth)) {
    print $row->email . "\n";
}
ibase_close ($dbh);
```

Voir aussi : [ibase_pconnect\(\)](#).

[10.29.2 ibase_pconnect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_connect](#) (string *database*, string *username* , string *password*)

Ouvre une connexion persistante à une base de données Interbase.

Voir aussi [ibase_connect\(\)](#).

[10.29.3 ibase_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_close](#) (int *connection_id*)

Ferme une connexion à une base de données Interbase. Cette fonction prend comme argument un identifiant de connexion id retourné par [ibase_connect\(\)](#).

[10.29.4 ibase_query](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_query](#) (int *link_identifieur* , string *query* , int *bind_args*)

Exécute une requête sur une base Interbase, et retourne un identifiant de résultat, à utiliser avec [ibase_fetch_row\(\)](#), [ibase_free_result\(\)](#) et/ou [ibase_free_query\(\)](#).

[10.29.5 ibase_fetch_row](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_fetch_row](#) (*int result_identifieur*)

Retourne la prochaine ligne spécifiée dans le résultat obtenu de [ibase_query\(\)](#).

[10.29.6 ibase_fetch_object](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_fetch_object](#)(int *result_id*)

Lit une ligne dans une base Interbase et la place dans un pseudo objet. Cette fonction prend comme argument un identifiant de résultat obtenu de [ibase_query\(\)](#) ou [ibase_execute\(\)](#).

```
$dbh = ibase_connect ($host, $username, $password);
$stmt = 'SELECT * FROM tblname';
$sth = ibase_query ($dbh, $stmt);
while ($row = ibase_fetch_object ($sth)) {
    print $row->email . "\n";
}
ibase_close ($dbh);
```

Voir aussi [ibase_fetch_row\(\)](#).

[10.29.7 ibase_free_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_free_result](#)(int *result_identifieur*)

Libère un résultat créé par [ibase_query\(\)](#).

[10.29.8 ibase_prepare](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_prepare](#)(int *link_identifieur* , string *query*)

Prépare une requête pour lier les paramètres (avec [ibase_bind\(\)](#)) et l'exécuter (avec [ibase_execute\(\)](#)).

[10.29.9 ibase_bind](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

[ibase_bind](#) (*int query*)

Lie les paramètres avec une requête précédemment préparée avec [ibase_prepare\(\)](#).

Note : [ibase_bind\(\)](#) ne fonctionne actuellement pas avec PHP4

[10.29.10 ibase_execute](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_execute](#)(*int query*)

Execute une requête préparée (et éventuellement liée) par [ibase_prepare\(\)](#) (et éventuellement [ibase_bind\(\)](#)).

[10.29.11 ibase_free_query](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

[ibase_free_query](#) (*int query*)

Libère la mémoire réservée par une requête préparée par [ibase_prepare\(\)](#).

[10.29.12 ibase_timefmt](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

[ibase_timefmt](#) (*string format*)

Fixe le format de date pour les prochaines requêtes.

Note : [ibase_timefmt\(\)](#) ne fonctionne pas sous PHP4.

[10.29.13 ibase_num_fields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_num_fields](#) (int *result_id*)

Retourne le nombre de lignes dans un résultat.

```
$dbh = ibase_connect ($host, $username, $password);
$stmt = 'SELECT * FROM tblname';
$stmt = ibase_query ($dbh, $stmt);
if ( ibase_num_rows($sth) > 0 ) {
while ($row = ibase_fetch_object ($sth)) {
print $row->email . "\n";
}
} else {
die("Aucun résultat");
}
ibase_close ($dbh);
```

Note : [ibase_timefmt\(\)](#) ne fonctionne pas encore sous PHP4.

[10.30 Fonctions ICAP](#)

[\[Notes en ligne\]](#)

Pour faire fonctionner ces fonctions, vous devez compiler PHP avec l'option `--with-icap`. Il vous faudra aussi la librairie icap. Récupérez la dernière version à <http://icap.chek.com/> et décompilez-la, puis installez la.

[10.30.1 icap_open](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

stream [icap_open](#) (string*calendar*, string*username*, string*password*, string*options*)

Retourn un flot ICAP en cas de succès, FALSE en cas d'erreur.

[icap_open\(\)](#) ouvre une connexion ICAP au serveur de calendrier *calendar*. Si le paramètre *options* est spécifié, passe *options* à la boîte aux lettres(????).

[10.30.2 icap_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [icap_close](#) (int *icap_stream*, int *flags*)

Ferme le flot ICAP *icap_stream*.

[10.30.3 icap_fetch_event](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [icap_fetch_event](#) (stream*icap_stream*, id*event id*, options*options*)

[icap_fetch_event\(\)](#) recherche un événement dans le calendrier spécifié par *id*.

Retourne un objet événement dont les attributs sont :

- int id – ID de l'événement.
- int public – TRUE si l'événement est public, FALSE si il est privé.
- string category – Catégorie de l'événement.
- string title – Titre de l'événement.
- string description – Description de l'événement.
- int alarm – Nombre de minutes avant d'envoyer une alerte pour cet événement.
- object start – Objet contenant une date et une heure.
- object end – Objet contenant une date et une heure.

Tous les objets de date et heure sont construits comme suit :

- int year – année
- int month – mois
- int mday – jour du mois
- int hour – heure
- int min – minutes
- int sec – secondes

[10.30.4 icap_list_events](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [icap_list_events](#) (stream *icap_stream*, datetime *begin_date*, datetime *end_date*)

Retourne un tableau d'identifiants d'événements, compris entre deux dates.

[icap_list_events\(\)](#) prend une date de début et une date de fin. Un tableau d'identifiants est retourné.

Tous les objets de date et heure sont construits comme suit :

- int year – année
- int month – mois
- int mday – jour du mois
- int hour – heure
- int min – minutes
- int sec – secondes

10.30.5 icap_store_event

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [icap_store_event](#)(int *icap_stream*, object *event*)

[icap_store_event\(\)](#) enregistre un événement dans un calendrier ICAP.

Un événement est un objet avec les attributs suivants :

- int id – ID de l'événement.
- int public – TRUE si l'événement est public, FALSE si il est privé.
- string category – Catégorie de l'événement.
- string title – Titre de l'événement.
- string description – Description de l'événement.
- int alarm – Nombre de minutes avant d'envoyer une alerte pour cet événement.
- object start – Objet contenant une date et une heure.
- object end – Objet contenant une date et une heure.

Tous les objets de date et heure sont construits comme suit :

- int year – année
- int month – mois
- int mday – jour du mois
- int hour – heure
- int min – minutes
- int sec – secondes

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

10.30.6 icap_delete_event

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [icap_delete_event](#)(int *uid*)

[icap_delete_event\(\)](#) efface l'événement d'identifiant *uid*.

Retourne TRUE.

10.30.7 icap_snooze

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [icap_snooze](#) (int *uid*)

[icap_snooze\(\)](#) éteint l'alarme de l'événement identifié par l'UID *uid*.

Retourne TRUE.

10.30.8 icap_list_alarms

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [icap_list_alarms](#) (stream *icap_stream*, datetime *alarm_date*)

Retourne un tableau d'identifiants, qui ont une alarme de prévue à la date *alarm_date*.

[icap_list_alarms\(\)](#) prend une date, et retourne un tableau d'identifiants.

Tous les objets de date et heure sont construits comme suit :

- int year – année
- int month – mois
- int mday – jour du mois
- int hour – heure
- int min – minutes
- int sec – secondes

10.31 Fonctions Informix

[\[Notes en ligne\]](#)

Le pilote d'accès à Informix pour Online (ODS) 7.x, SE 7.x et Universal Server (IUS) 9.x est implémenté dans "functions/ifx.ec" et "functions/php3_ifx.h". Le support ODS 7.x est plutôt complet, et accepte les colonnes de type BYTE et TEXT. Le support IUS 9.x est partiellement fini, de nouveaux types sont disponibles, mais SLOB et CLOB sont toujours en développement.

Note : Avant que vous ne lanciez le script "configure", assurez vous que la variable d'environnement "INFORMIXDIR" a été correctement paramétrée.

Le script de configuration va détecter automatiquement les bibliothèques disponibles, et inclure les dossiers si vous lancez le script avec l'option "configure --with_informix=yes". Vous pouvez ignorer cette détection en spécifiant "IFX_LIBDIR", "IFX_LIBS" et "IFX_INCDIR" dans votre environnement. Le script de configuration va aussi essayer de détecter la version de votre serveur Informix. Il modifiera alors la condition de compilation "HAVE_IFX_IUS" si votre serveur Informix est d'une version plus récente que 9.00.

Note : Assurez vous que les variables d'environnement INFORMIXDIR et INFORMIXSERVER sont accessibles au pilote PHP, et que le dossier bin INFORMIX est aussi dans la variable PATH. Vous pouvez le voir en lançant un script qui contient un appel à [phpinfo\(\)](#) avant que vous ne commenciez à tester. La fonction [phpinfo\(\)](#) affiche une liste des variables d'environnement. Cela fonctionne aussi bien en mode mod_php, qu'en mode CGI. Il vous faudra fixer les valeurs dans le script de démarrage d'Apache. Les "Informix shared libraries" doivent aussi être accessibles au chargement (vérifiez LD_LIBRARY_PATH

ou `ld.so.conf/ldconfig`).

Note : Les objets de type BLOBs sont normalement gérés par des identifiants de BLOB. Les requêtes de sélection retournent un identifiant de BLOB pour chaque colonne de type BYTE et TEXT. Vous pouvez en lire le contenu, avec des commandes de types `"string_var = ifx_get_blob($BLOB_id);"` ; si vous souhaitez ramener le BLOB en mémoire (avec: `"ifx_blobinfile_mode(0);"`). Si vous préférez recevoir le contenu d'une colonne BLOB dans un fichier, utilisez [ifx_blobinfile_mode\(\)](#), et `ifx_get_blob($BLOB_id)` vous retournera le nom du fichier. Utilisez les fonctions habituelles d'accès aux fichiers pour lire son contenu.

Pour les requêtes INSERT/UPDATE, vous devez créer les identifiants de BLOB par vous même, avec la fonction [ifx_create_blob\(\)](#). Puis, vous placez l'identifiant de BLOB dans un tableau, et remplacez la colonne par un point d'interrogation. Pour les UPDATE/INSERT, vous êtes responsable du contenu du BLOB, avec la fonction [ifx_update_blob\(\)](#).

Le comportement par défaut des colonnes de type BLOB peut être modifié en affectant de nouvelles valeurs aux variables de configuration (même à la volée) :

Variable de configuration : `ifx.textasvarchar`

Variable de configuration : `ifx.byteasvarchar`

Fonctions à utiliser lors de l'exécution :

`ifx_textasvarchar(0)` : Utilise l'identifiant de BLOB avec des colonnes de type TEXT, dans les requêtes SELECT

`ifx_byteasvarchar(0)` : Utilise l'identifiant de BLOB avec des colonnes de type BYTE, dans les requêtes SELECT

`ifx_textasvarchar(1)` : Retourne les colonnes de type TEXT sous la forme de VARCHAR, sans utiliser les identifiants de BLOB dans les requêtes SELECT.

`ifx_byteasvarchar(1)` : Retourne les colonnes de type BYTE sous la forme de VARCHAR, sans utiliser les identifiants de BLOB dans les requêtes SELECT.

Variable de configuration : `ifx.BLOBinfile`

Fonctions à utiliser lors de l'exécution :

`ifx_blobinfile_mode(0)` : Retourne les colonnes de type BYTE en mémoire, l'identifiant de BLOB vous donnera accès au contenu.

`ifx_blobinfile_mode(1)` : Retourne les colonnes de type BYTE dans un fichier, l'identifiant de BLOB vous donnera accès au nom de ce fichier.

En affectant la valeur de 1 à `ifx_text/byteasvarchar`, vous pouvez utiliser les colonnes de type TEXT et BYTE dans les requêtes SELECT comme des champs VARCHAR (mais plus long). Etant donné la gestion des chaînes par PHP, cette technique conserve les données binaires. Les données retournées peuvent contenir n'importe quoi, et vous êtes responsable de la bonne manipulation de ces valeurs.

En affectant la valeur de 1 à [ifx_blobinfile_mode\(\)](#), utilisez le nom de fichier retourné par [ifx_get_blob\(\)](#) pour accéder au contenu du BLOB. Notez bien que vous êtes tenu responsable de l'effacement des fichiers temporaires, créés par Informix. Chaque nouvelle ligne lue sur le serveur va créer un nouveau fichier temporaire, pour chaque colonne de type BYTE.

L'emplacement des fichiers temporaire peut être modifié, gr'ce à la variable `"blobdir"`, (par défaut, `."`, c'est à dire, le dossier courant). Une valeur telle que `BLOBdir="tmpBLOB"` simplifiera le nettoyage des fichiers temporaires, accidentellement oubliés (les noms commencent tous par `"blb"`).

Note : Elle peut être mise en place avec la variable de configuration.

`ifx.charasvarchar` : avec la valeur 1, les espaces de fin de champs seront automatiquement supprimés.

Note : Lorsque la variable de configuration `ifx.nullformat` (ou que la fonction [ifx_nullformat\(\)](#)) est à un, les colonnes contenant la valeur NULL retourneront la chaîne "NULL", et sinon, retourneront une chaîne vide. Cela vous permet de faire la différence entre les colonnes vides et celle qui contiennent la valeur NULL.

10.31.1 ifx_connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_connect](#)(string *database* , string *userid* , string *password*)

Retourne un identifiant de connexion, en cas de succès, et FALSE sinon.

[ifx_connect\(\)](#) établit une connexion à un serveur Informix. Tous les arguments sont optionnels, et, si ils viennent à manquer, les valeurs par défaut seront prises dans le fichier [7.1 Le fichier de configuration](#). (ifx.default_host pour l'hôte par défaut) (Les bibliothèques Informix utiliseront la variable d'environnement `$INFORMIXSERVER` si ifx.default_host n'est pas définie). ifx.default_user pour l'utilisateur, et ifx.default_password comme mot de passe (si aucun n'a été défini).

Si un deuxième appel à [ifx_connect\(\)](#) est fait avec les mêmes arguments, l'identifiant de connexion déjà ouvert sera retourné.

Le lien avec le serveur sera fermé dès que le script se termine, ce qui fait qu'il n'est pas nécessaire de terminer les connexions avec [ifx_close\(\)](#).

Voir aussi [ifx_pconnect\(\)](#) et [ifx_close\(\)](#). *Connection à un serveur Informix*

```
$conn_id = ifx_pconnect ("mydb@ol_srv1", "imyslf", "mypassword");
```

[10.31.2 ifx_pconnect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_pconnect](#)(string *database* , string *userid* , string *password*)

Retourne un identifiant positif de connexion Informix, ou FALSE, en cas d'erreur.

[ifx_pconnect\(\)](#) se comporte de manière très similaire à [ifx_connect\(\)](#) avec deux différences importantes : Cette fonction se comporte exactement comme [ifx_connect\(\)](#) lorsque PHP n'est pas un module Apache. Lors de la connexion, la fonction va chercher une connexion déjà ouverte avec le même hôte, le même nom d'utilisateur, et le même mot de passe. Si elle en trouve une, elle retournera un identifiant de cette connexion, au lieu d'en ouvrir une nouvelle.

Deuxièmement, la connexion au serveur SQL ne sera pas automatiquement refermée à la fin de l'exécution du script. Au contraire, le lien va rester ouvert ([ifx_close\(\)](#) ne fermera pas les connexions établies avec [ifx_pconnect\(\)](#)).

Ainsi, ce type de lien est appelé 'persistant'.

Voir aussi: [ifx_connect\(\)](#).

[10.31.3 ifx_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_close](#)(int *link_identifier*)

Retourne toujours TRUE.

[ifx_close\(\)](#) ferme le lien au serveur de données Informix associé à l'identifiant de connexion *link_identifier*. Si l'identifiant du lien n'est pas spécifié, la dernière connexion est utilisée.

Notez qu'il n'est généralement pas besoin d'appeler cette fonction, car les connexions non persistantes seront automatiquement fermées.

[ifx_close\(\)](#) ne peut pas fermer une connexion ouverte avec [ifx_pconnect\(\)](#).

Voir aussi: [ifx_connect\(\)](#) et [ifx_pconnect\(\)](#). *Fermer une connexion Informix*

```
$conn_id = ifx_connect ("mydb@ol_srv", "itsme", "mypassword");
... some queries and stuff ...
ifx_close($conn_id);
```

10.31.4 ifx_query

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int ifx_query (string *query*, int *link_identifieur* , int *cursor_type* , mixed *BLOBidarray*)

Retourne un identifiant positif de résultat Informix en cas de succès, et FALSE en cas d'erreur.

L'entier de type "identifiant de résultat" est utilisé par d'autres fonctions pour lire les résultats. Pour un exemple, reportez vous à [ifx_affected_rows\(\)](#) pour connaître le nombre de lignes affectées.

[ifx_query\(\)](#) envoie une requête au serveur actif courant, associé à l'identifiant de connexion *link_identifieur*. Si *link_identifieur* n'est pas fourni, la dernière connexion ouverte sera utilisée. Si aucune connexion n'a été ouverte, [ifx_query\(\)](#) va essayer d'en créer une, en appelant [ifx_connect\(\)](#).

Exécute la requête *query* sur la connexion *conn_id*. Pour les requêtes de type SELECT, un pointeur est déclaré, et ouvert. L'option *cursor_type* permet de choisir le type de pointeur, "scroll" et/ou "hold".

cursor_type accepte les deux valeurs séparées, et leur combinaison. Les requêtes d'autre type sont à exécution immédiate.

Le nombre de ligne affectées (estimé ou exact) est enregistré, pour être lu avec la fonction [ifx_affected_rows\(\)](#).

Si vous avez une colonne de type BLOB (BYTE ou TEXT) dans une requête de modification, vous pouvez passer un paramètre *BLOBidarray* qui contiendra les identifiants des BLOB à modifier, et vous devrez remplacer cette colonne par un point d'interrogation (?) dans la requête.

Si le contenu d'une colonne de type TEXT (ou BYTE) vous pouvez aussi utiliser les fonctions [ifx_textasvarchar\(\)](#) et [ifx_byteasvarchar\(\)](#). Cela vous permettra d'utiliser les colonnes TEXT (ou BYTE) comme des colonnes de type VARCHAR (mais plus long, tout de même), et vous n'aurez pas besoin de l'identifiant de BLOB.

Avec les fonctions [ifx_textasvarchar\(\)](#) et [ifx_byteasvarchar\(\)](#) (valeurs par défaut), les requêtes SELECT retourneront des identifiants de BLOB. Cet identifiant peut être une chaîne ou un fichier, suivant la configuration (voir plus loin).

Voir aussi : [ifx_connect\(\)](#). Afficher toutes les lignes de la table "ordres" sous la forme html *Afficher toutes les lignes de la table "ordres" sous la forme html*

```
ifx_textasvarchar(1); // Utilisation du mode "text mode" pour les BLOBs
$res_id = ifx_query("select * from orders", $conn_id);
if (! $res_id) {
printf("Impossible de selectionner des lignes dans : %s\n<br>%s<br>\n", ifx_error());
ifx_errormsg();
die;
}
ifx_htmltbl_result($res_id, "border=\\"1\\");
ifx_free_result($res_id);
```

Insertion de valeurs dans la table "catalogue"

```

        // créer un identifiant de BLOB pour une colonne de type BYTE et une de typ
$textid = ifx_create_blob(0, 0, "Text column in memory");
$byteid = ifx_create_blob(1, 0, "Byte column in memory");
        // store BLOB id's in a BLOBid array
$BLOBidarray[] = $textid;
$BLOBidarray[] = $byteid;
        // launch query
$query = "insert into catalog (stock_num, manu_code, " .
        "cat_descr,cat_picture) values(1,'HRO',?,?)";
$res_id = ifx_query($query, $conn_id, $BLOBidarray);
if (! $res_id) {
    ... erreur ...
}
        // libération du résultat
ifx_free_result($res_id);

```

10.31.5 ifx_prepare

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_prepare](#)(string *query*, int *conn_id*, int *cursor_def*, mixed *BLOBidarray*)

Retourne un entier identifiant de résultat *result_id* à utiliser avec [ifx_do\(\)](#). Modifie la valeur de *affected_rows*, pour accès ultérieur avec [ifx_affected_rows\(\)](#).

Prépare la requête *query* sur la connexion *conn_id*. Pour les requêtes de type "select-type" un pointeur de résultat est déclaré et ouvert. L'option *cursor_type* permet de choisir le type de pointeur : "scroll" et/ou "hold". Les valeurs peuvent être combinées ensemble (IFX_SCROLL, IFX_HOLD).

Le nombre de ligne affectés (estimé ou exact) est enregistré, pour être lu avec la fonction [ifx_affected_rows\(\)](#).

Si vous avez une colonne de type BLOB (BYTE ou TEXT) dans une requête de modification, vous pouvez passer un paramètre *BLOBidarray* qui contiendra les identifiants des BLOB à modifier, et vous devrez remplacer cette colonne par un point d'interrogation (?) dans la requête.

Si le contenu d'une colonne de type TEXT (ou BYTE) vous pouvez aussi utiliser les fonctions [ifx_textasvarchar\(\)](#) et [ifx_byteasvarchar\(\)](#). Cela vous permettra d'utiliser les colonnes TEXT (ou BYTE) comme des colonnes de type VARCHAR (mais plus long, tout de même), et vous n'aurez pas besoin de l'identifiant de BLOB.

Avec les fonctions [ifx_textasvarchar\(\)](#) et [ifx_byteasvarchar\(\)](#) (valeurs par défaut), les requêtes SELECT retourneront des identifiants de BLOB. Cet identifiant peut être une chaîne ou un fichier, suivant la configuration (voir plus loin).

Voir aussi [ifx_do\(\)](#).

10.31.6 ifx_do

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_do](#)(int *result_id*)

Retourne TRUE en cas de succès, FALSE en cas d'erreur.

Exécute une requête qui a déjà été préparée, ou crée un pointeur pour cela.

Ne libère pas *result_id* en cas d'erreur.

De plus, elle fixe la valeur de *result_id* pour accès ultérieur par [ifx_affected_rows\(\)](#).
Voir aussi : [ifx_prepare\(\)](#). Il y a un exemple.

[10.31.7 ifx_error](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ifx_error](#)

Les codes d'erreur Informix (SQLSTATE SQLCODE) formaté; comme suit :

x [SQLSTATE = aa bbb SQLCODE=cccc]

avec x = space : aucune erreur

E : erreur

N : il n'y a plus d'informations

W : Alerte

? : Undéfinie

Si le caractère vaut autre chose qu'un espace, SQLSTATE et SQLCODE décrit l'erreur avec plus de détails.

Reportez vous au manuel Informix pour trouver la description de SQLSTATE et SQLCODE

Retourne une chaîne avec un caractères, décrivant le résultat général de la commande, et aussi SQLSTATE et SQLCODE associé à la plus récente requête SQL exécutée. Le format de la chaîne est "(char

[SQLSTATE=(deux chiffres) (trois chiffres) SQLCODE=(un chiffre)]". Le premier caractère peut être ' ' (espace) (succès), 'W' (Alerte), 'E' (une erreur est survenue durant le traitement) ou 'N' (aucune donnée de retour).

Voir aussi: [ifx_errormsg\(\)](#).

[10.31.8 ifx_errormsg](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ifx_errormsg](#)(int *errorcode*)

Retourne le plus récent message d'erreur ou, lorsque l'option *errorcode* est présent, le message d'erreur associé à *errorcode*.

Voir aussi: [ifx_error\(\)](#).

```
printf("%s\n<br>", ifx_errormsg(-201));
```

[10.31.9 ifx_affected_rows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_affected_rows](#)(int *result_id*)

result_id est un identifiant valide de résultat retourné par [ifx_query\(\)](#) ou [ifx_prepare\(\)](#).

Retourne le nombre de lignes affectées par la requête associée à *result_id*.

Pour les INSERT, UPDATE et DELETE, ce nombre est le nombre exact de lignes affectées (sqlerrd[2]). Pour les SELECT, ce n'est qu'une estimation (sqlerrd[0]). Ne vous y fiez pas.

Cette fonction est très pratique après [ifx_prepare\(\)](#) pour limiter la taille des résultats.

Voir aussi : [ifx_num_rows\(\)](#).

Nombre de lignes affectées

```

$rid = ifx_prepare ("select * from emp
where name like " . $name, $connid);
if (! $rid) {
    ... error ...
}
$rowcount = ifx_affected_rows ($rid);
if ($rowcount > 1000) {
printf ("Trop de lignes trouvées (%d)\n<br>", $rowcount);
die ("Ressayez avec une autre requête. <br>\n");
}

```

[10.31.10 ifx_getsqlca](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ifx_getsqlca](#) (int *result_id*)

result_id est un identifiant valide de résultat retourné par [ifx_query\(\)](#) ou [ifx_prepare\(\)](#).

Retourne une pseudo-ligne (tableau associatif) avec `sqlca.sqlerrd[0]` à `sqlca.sqlerrd[5]` après la requête associée *result_id*.

Pour les requêtes INSERT, UPDATE et DELETE, les valeurs retournées sont celles fixées par le serveur après avoir exécuté la requête. Cela donne accès au nombre de ligne affectées, ainsi qu'au numéro de série d'insertion. Pour les requêtes de type SELECT, les valeurs retournées sont celles qui ont été préparées. Utiliser cette fonction économise l'exécution d'une requête "select dbinfo('sqlca.sqlerrdx')", étant donné qu'elle retourne les valeurs qui ont été sauvées par le pilote ifx driver au moment approprié.

Lire les valeurs de sqlca.sqlerrd[x]

```

/* On suppose que la première colonne d'une table 'quelconque' est un numéro de série */
$qid = ifx_query("insert into sometable values(0, '2nd column', 'another column' ", $connid);
if (! $qid) {
    ... error ...
}
$sqlca = ifx_getsqlca ($qid);
$serial_value = $sqlca["sqlerrd1"];
echo "Le numéro de série de la valeur insérée est : " . $serial_value<br>\n";

```

[10.31.11 ifx_fetch_row](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ifx_fetch_row](#) (int *result_id*, mixed *position*)

Retourne un tableau associatif qui contient la ligne retournée, ou FALSE si il ne reste plus de lignes à lire, ou si il a eu une erreur.

Les colonnes de types BLOB sont retournées sous la forme d'un identifiant à utiliser avec [ifx_get_blob\(\)](#) à moins que vous n'ayez utilisé la fonction [ifx_textasvarchar\(\)](#) ou [ifx_byteasvarchar\(\)](#), et dans ce cas, les BLOBs seront retournés sous forme de chaîne. Retourne FALSE en cas d'erreur.

result_id est un identifiant valide de résultat, retourné par [ifx_query\(\)](#) ou [ifx_prepare\(\)](#) (Requêtes SELECT seulement !).

position est un paramètre optionnel, pour une opération de lecture d'informations sur un pointeur de type "scroll": "NEXT", "PREVIOUS", "CURRENT", "FIRST", "LAST" ou encore un nombre. Si vous spécifier un nombre, la ligne d'index absolu sera retournée. Ce paramètre est optionnel, et ne fonctionne qu'avec les pointeurs de type "scroll".

[ifx_fetch_row\(\)](#) retourne une ligne de données d'un résultat associé à l'identifiant de résultat *result_id*. La ligne est retournée sous la forme d'un tableau associatif.

Les appels ultérieurs à [ifx_fetch_row\(\)](#) retourneront la ligne suivante, ou FALSE si il n'y a plus de ligne.

Informix fetch rows

```
$rid = ifx_prepare ("select * from emp where name like " . $name,
                  $connid, IFX_SCROLL);
if (! $rid) {
    ... error ...
}
$rowcount = ifx_affected_rows($rid);
if ($rowcount > 1000) {
    printf ("Trop de lignes dans le résultats. (%d)\n<br>", $rowcount);
    die ("Recommencez votre requête. <br>\n");
}
if (! ifx_do ($rid)) {
    ... erreur ...
}
$row = ifx_fetch_row ($rid, "NEXT");
while (is_array($row)) {
    for(reset($row); $fieldname=key($row); next($row)) {
        $fieldvalue = $row[$fieldname];
        printf ("%s = %s,", $fieldname, $fieldvalue);
    }
    printf("\n<br>");
    $row = ifx_fetch_row ($rid, "NEXT");
}
ifx_free_result ($rid);
```

10.31.12 ifx_htmltbl_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_htmltbl_result](#)(int *result_id*, string *html_table_options*)

Lit toutes les lignes d'un tableau, et la met sous la forme d'un tableau HTML, ou FALSE en cas d'erreur. Affiche les lignes avec des balises HTML. Le second argument permet de modifier les options de table.

Affichage sous la forme d'une table HTML

```
$rid = ifx_prepare ("select * from emp where name like " . $name,
                  $connid, IFX_SCROLL);
if (! $rid) {
    ... error ...
}
$rowcount = ifx_affected_rows ($rid);
if ($rowcount > 1000) {
    printf ("Trop de lignes dans le résultat : (%d)\n<br>", $rowcount);
    die ("Recommencez votre requête <br>\n");
}
if (! ifx_do($rid)) {
    ... erreur ...
}
ifx_htmltbl_result ($rid, "border=\"2\"");
ifx_free_result($rid);
```

[10.31.13 ifx_fieldtypes](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ifx_fieldtypes](#) (int *result_id*)

Retourne un tableau associatif avec les noms des champs comme clés, et les types SQL comme valeur. En cas d'erreur, retourne FALSE.

Nom de champs et type SQL.

```

$types = ifx_fieldtypes ($resultid);
if (! isset ($types)) {
    ... error ...
}
for ($i = 0; $i < count($types); $i++) {
    $fname = key($types);
    printf("%s :\t type = %s\n", $fname, $types[$fname]);
    next($types);
}

```

[10.31.14 ifx_fieldproperties](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ifx_fieldproperties](#) (int *result_id*)

Retourne un tableau associatif avec les nom des champs comme clé, et les données de propriétés des champs comme valeur. Retourne FALSE en cas d'erreur.

Retourne les propriétés Informix SQL pour tous les champs d'une requête, sous la forme d'un tableau associatif. Les propriétés sont présentées sous la forme : "SQLTYPE;longueur;précision;échelle;ISNULLABLE" avec SQLTYPE qui représente le type de données Informix tel que "SQLVCHAR"etc÷. et ISNULLABLE = "Y" ou "N" (le champs peut contenir NULL ou pas : Oui ou Non).

Informix SQL fieldproperties

```

$properties = ifx_fieldtypes ($resultid);
if (! isset($properties)) {
    ... error ...
}
for ($i = 0; $i < count($properties); $i++) {
    $fname = key ($properties);
    printf ("%s:\t type = %s\n", $fname, $properties[$fname]);
    next ($properties);
}

```

[10.31.15 ifx_num_fields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_num_fields](#) (int *result_id*)

Retourne le nombre de colonnes dans la requête *result_id* ou FALSE en cas d'erreur.

Après avoir préparé ou exécuté une requête, cette fonction retourne le nombre de colonne dans la requête.

[10.31.16 ifx_num_rows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_num_rows](#)(int *result_id*)

Compte le nombre de ligne déjà lues dans le résultat *result_id* après [ifx_query\(\)](#) ou [ifx_do\(\)](#).

[10.31.17 ifx_free_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_free_result](#)(int *result_id*)

Libère les ressources prises par le résultat *result_id*. Retourne FALSE en cas d'erreur.

[10.31.18 ifx_create_char](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_create_char](#)(string *param*)

Crée un objet char. *param* sera le contenu de l'objet.

[10.31.19 ifx_free_char](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_free_char](#)(int *bid*)

Supprime l'objet char *bid*. Retourne FALSE en cas d'erreur, et sinon TRUE.

[10.31.20 ifx_update_char](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_update_char](#)(int *bid*, string *content*)

Modifie le contenu de l'objet char repéré par son identifiant *bid*. *content* est une chaîne avec les nouvelles données. Retourne. FALSE en cas d'erreur, et sinon, TRUE.

[10.31.21 ifx_get_char](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_get_char](#)(int *bid*)

Retourne le contenu de l'objet associé à l'identifiant *bid*.

[10.31.22 ifx_create_blob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_create_blob](#)(int *type*, int *mode*, string *param*)

Crée un objet BLOB.

type: 1 = TEXT, 0 = BYTE

mode: 0 = L'objet BLOB place le contenu en mémoire ; 1 = L'objet BLOB place le contenu dans un fichier.

param: Si mode = 0: pointeur du contenu, si mode = 1: pointeur vers un fichier.

Retourne FALSE en cas d'erreur, et sinon, un identifiant de BLOB.

[10.31.23 ifx_copy_blob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_copy_blob](#)(int *bid*)

Retourne FALSE en cas d'erreur, et sinon, l'identifiant du nouvel objet.

[10.31.24 ifx_free_blob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_free_blob](#)(int *bid*)

Supprime l'objet BLOB *bid*. Retourne FALSE en cas d'erreur, et sinon TRUE.

[10.31.25 ifx_get_blob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_get_blob](#)(int *bid*)

Retourne le contenu de l'objet BLOB associé à *bid*.

[10.31.26 ifx_update_blob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

[ifx_update_blob](#) (int *bid*, string *content*)

Modifie le contenu de l'objet BLOB repéré par son identifiant *bid*. *content* est une chaîne contenant les nouvelles données. Retourne FALSE en cas d'erreur, et sinon, TRUE.

[10.31.27 ifx_blobinfile_mode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ifx_blobinfile_mode](#)(int *mode*)

Choisi le mode par défaut des objets BLOB pour toutes les requêtes SELECT. Mode "0" chargera les BLOB

de type Byte en mémoire ; Mode "1" sauvera les BLOB de type Byte dans un fichier.

[10.31.28 ifx_textasvarchar](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ifx_textasvarchar](#)(int *mode*)

Choisi le mode par défaut des objets TEXT. Le mode "0" retournera un identifiant de BLOB et le mode "1" retourne le BLOB sous la forme d'un (gros) varchar.

[10.31.29 ifx_byteasvarchar](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ifx_byteasvarchar](#)(int *mode*)

Choisi le mode par défaut des objets BYTE. Le mode "0" retournera l'identifiant de BLOB, et le mode "1" retournera le contenu du text sous la forme d'un VARCHAR.

[10.31.30 ifx_nullformat](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ifx_nullformat](#)(int *mode*)

Choisi le mode par défaut de lecture des valeurs. Le mode "0" retourne "", et le mode "1" retourne "NULL".

[10.31.31 ifxus_create_slob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_create_slob](#)(int *mode*)

Crée un objet SLOB et l'ouvre. Les modes valides sont : 1 = LO_RDONLY, 2 = LO_WRONLY, 4 = LO_APPEND, 8 = LO_RDWR, 16 = LO_BUFFER, 32 = LO_NOBUFFER -> ou une combinaison des précédents. Vous pouvez aussi utiliser les constantes suivantes : IFX_LO_RDONLY, IFX_LO_WRONLY etc. Retourne FALSE en cas d'erreur, et sinon, l'identifiant de l'objet SLOB.

[10.31.32 ifx_free_slob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_free_slob](#)(int *bid*)

Supprime un objet SLOB. *bid* est l'identifiant de l'objet SLOB. Retourne FALSE en cas d'erreur, et sinon TRUE.

[10.31.33 ifxus_close_slob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_close_slob](#)(int *bid*)

Ferme l'objet SLOB représenté par son identifiant *bid*. Retourne FALSE en cas d'erreur, et sinon, TRUE.

[10.31.34 ifxus_open_slob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_open_slob](#)(long *bid*, int *mode*)

Ouvre un objet SLOB. *bid* est un identifiant d'objet SLOB. Les modes valides sont : 1 = LO_RDONLY, 2 = LO_WRONLY, 4 = LO_APPEND, 8 = LO_RDWR, 16 = LO_BUFFER, 32 = LO_NOBUFFER -> ou une combinaison des valeurs précédentes. Retourne FALSE en cas d'erreur, et sinon, l'identifiant du nouvel objet.

[10.31.35 ifxus_tell_slob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_tell_slob](#)(long *bid*)

Retourne le fichier courant, ou la position courante d'un objet SLOB ouvert. *bid* est un identifiant d'objet SLOB. Retourne FALSE en cas d'erreur, et sinon, la position du pointeur de fichier.

[10.31.36 ifxus_seek_slob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_seek_slob](#)(long *bid*, int *mode*, long *offset*)

Fixe le fichier courant, ou la position du pointeur de fichier, pour un objet SLOB ouvert. *bid* est un identifiant d'objet SLOB. Les modes valides sont : 0 = LO_SEEK_SET, 1 = LO_SEEK_CUR, 2 = LO_SEEK_END et *offset* est un octet d'offset. Retourne FALSE en cas d'erreur, et sinon, la position du pointeur de fichier.

[10.31.37 ifxus_read_slob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_read_slob](#)(long *bid*, long *nbytes*)

Lit *nbytes* octets de l'objet SLOB *bid*. *bid* est un identifiant d'objet SLOB existant, et *nbytes* est le nombre d'octets à lire. Retourne FALSE en cas d'erreur, et sinon, une chaîne de caractères.

[10.31.38 ifxus_write_slob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_write_slob](#)(long *bid*, string *content*)

Ecrit une chaîne dans un objet SLOB. *bid* est un identifiant d'objet SLOB et *content* sont les données à écrire. Retourne FALSE en cas d'erreur, et sinon, le nombre d'octets écrits.

10.32 Images

[\[Notes en ligne\]](#)

Vous pouvez utiliser les fonctions PHP pour obtenir les tailles des images aux formats JPEG, GIF, et PNG, et si vous avez la librairie GD (disponible à <http://www.boutell.com/gd/>) vous pourrez aussi créer et manipuler ces images.

10.32.1 GetImageSize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [getimagesize](#)(string *filename*, array *imageinfo*)

[getimagesize\(\)](#) va déterminer la taille des images de type *GIF*, *JPG* ou *PNG* et en retourner les dimensions avec le type d'image, et une chaîne type "height/width", à placer dans une balise *HTML* ou IMG normale. Retourne un tableau de 4 éléments. L'index 0 contient la largeur. L'index 1 contient la longueur. L'index 2 contient le type de l'image : 1 = GIF, 2 = JPG, 3 = PNG. L'index 3 contient la chaîne à placer dans les balises HTML : "height=xxx width=xxx". *GetImageSize*

```
<?php $size = GetImageSize("img/flag.jpg"); ?>
<IMG SRC="img/flag.jpg" <?php echo $size[3]; ?>>
```

Le paramètre optionnel *imageinfo* permet d'extraire des informations supplémentaires du fichier image. Actuellement, cette option va retourner différents marqueurs *JPG APP* dans un tableau associatif. Certains programmes utilisent ces marqueur APP pour préciser les informations dans les balises HTML. Un marqueur commun est le marqueur APP13, décrit à <http://www.xe.net/iptc/>. Vous pouvez utiliser la fonction [iptcparse\(\)](#) pour analyser ce marqueur, et obtenir des informations intelligibles. *GetImageSize qui retourne IPTC*

```
<?php
    $size = GetImageSize("testimg.jpg",info);
    if (isset($info["APP13"])) {
        $iptc = iptcparse($info["APP13"]);
        var_dump($iptc);
    }
?>
```

Note : Cette fonction ne requiert par la bibliothèque GD.

10.32.2 ImageArc

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagearc](#)(int *im*, int *cx*, int *cy*, int *w*, int *h*, int *s*, int *e*, int *col*)

[imagearc\(\)](#) dessine une ellipse partielle, centrée sur *cx*, *cy*, (le coin en haut à gauche est l'origine (0,0)) dans

l'image référencée par *im*. *w* et *h* spécifient la largeur et la hauteur de l'ellipse, tandis que le début et la fin de l'arc sont donnés en degrés, par les arguments *s* et *e*.

[10.32.3 ImageChar](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagechar](#) (int *im*, int *font*, int *x*, int *y*, string *c*, int *col*)

[imagechar\(\)](#) dessine le premier caractère de la chaîne *c* dans l'image *id* avec le coin supérieur gauche placé à la position *x,y* (le coin en haut à gauche est l'origine (0,0)) avec la couleur *col*. Si la police est 1, 2, 3, 4 ou 5, une police intégrée sera utilisée (plus le chiffre est grand, plus grande est la police).0

Voir aussi [imageloadfont\(\)](#).

[10.32.4 ImageCharUp](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecharup](#) (int *im*, int *font*, int *x*, int *y*, string *c*, int *col*)

[imagecharup\(\)](#) dessine le premier caractère de la chaîne *c* dans l'image *id* avec le coin supérieur gauche placé à la position *x,y* (le coin en haut à gauche est l'origine (0,0)), avec la couleur *col*. Si la police est 1, 2, 3, 4 ou 5, une police intégrée sera utilisée (plus le chiffre est grand, plus grande est la police).

Voir aussi [imageloadfont\(\)](#).

[10.32.5 ImageColorAllocate](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecolorallocate](#) (int *im*, int *red*, int *green*, int *blue*)

[imagecolorallocate\(\)](#) retourne un identifiant de couleur, représentant la couleur composée avec les couleurs RGB (*red*, *green*, *blue*). L'argument *im* est le résultat de la fonction [imagecreate\(\)](#). [imagecolorallocate\(\)](#) doit être appelée pour créer chaque couleur qui sera représentée par *im*.

```
$white = ImageColorAllocate($im, 255,255,255);
$black = ImageColorAllocate($im, 0,0,0);
```

[10.32.6 ImageColorDeAllocate](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecolordeallocate](#) (int *im* , int *index*)

[imagecolordeallocate\(\)](#) désalloue une couleur précédemment allouée avec la fonction [imagecolorallocate\(\)](#).

```
$white = ImageColorAllocate($im, 255, 255, 255);
ImageColorDeAllocate($im, $white);
```

[10.32.7 ImageColorAt](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecolorat](#) (int *im*, int *x*, int *y*)

[imagecolorat\(\)](#) retourne l'index de la couleur du pixel situé aux coordonnées (*x*, *y*), dans l'image *im*. Voir aussi [imagecolorset\(\)](#) et [imagecolorsforindex\(\)](#).

[10.32.8 ImageColorClosest](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecolorclosest](#) (int *im*, int *red*, int *green*, int *blue*)

[imagecolorclosest\(\)](#) retourne l'index de la couleur de la palette qui est la plus proche de la valeur RGB passée. La "distance" entre la couleur souhaitée et les couleurs de la palette est calculée en considérant l'espace RGB comme un espace à 3 dimensions. Voir aussi [imagecolorexact\(\)](#).

[10.32.9 ImageColorExact](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecolorexact](#) (int *im*, int *red*, int *green*, int *blue*)

[imagecolorexact\(\)](#) retourne l'index de la couleur spécifiée dans la palette de l'image *im*. Si la couleur n'existe pas dans cette palette, retourne -1. Voir aussi [imagecolorclosest\(\)](#).

[10.32.10 ImageColorResolve](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecolorresolve](#) (int *im*, int *red*, int *green*, int *blue*)

[imagecolorresolve\(\)](#) retourne un index de couleur à tous les coups. Soit il arrive à trouver la couleur demandée dans la palette, soit il recherche la couleur la plus proche. Voir aussi [imagecolorclosest\(\)](#).

[10.32.11 ImageGammaCorrect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagegammacorrect](#) (int *im* , double *inputgamma* , double *outputgamma*)

[imagegammacorrect\(\)](#) applique une correction gamma au flot d'image GD *im*. Le facteur d'entrée est *inputgamma*, et le facteur de sortie est *outputgamma*.

10.32.12 ImageColorSet

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [imagecolorset](#)(int *im*, int *index*, int *red*, int *green*, int *blue*)

[imagecolorset\(\)](#) permet d'attribuer à un index d'une palette une couleur spécifique. C'est une fonction très pratique pour effectuer du remplissage de couleur sans le faire réellement.

Voir aussi [imagecolorat\(\)](#).

10.32.13 ImageColorsForIndex

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imagecolorsforindex](#)(int *im*, int *index*)

[imagecolorsforindex\(\)](#) retourne un tableau associatif avec les couleur rouge (red) , vert (green), bleu (blue) qui contiennent les valeurs de la couleur correspondante.

Voir aussi [imagecolorat\(\)](#) et [imagecolorexact\(\)](#).

10.32.14 ImageColorsTotal

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecolorstotal](#)(int *im*)

[imagecolorstotal\(\)](#) retourne le nombre de couleur de la palette.

Voir aussi [imagecolorat\(\)](#) et [imagecolorsforindex\(\)](#).

10.32.15 ImageColorTransparent

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecolortransparent](#)(int *im*, int *col*)

[imagecolortransparent\(\)](#) permet de choisir la couleur transparente d'une image, et de lui donner la valeur de col. *im* est un identifiant d'image, retourné par [imagecreate\(\)](#) et *col* est un identifiant de couleur retourné par [imagecolorallocate\(\)](#).

L'identifiant de la nouvelle (ou courante) couleur transparente est retourné.

10.32.16 ImageCopy

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecopy](#)(int *dst_im* , int *src_im* , int *dst_x* , int *dst_y* , int *src_x* , int *src_y* , int *src_w* , int *src_h*)

Copie une partie de l'image *src_im* sur l'image de destination *dst_im*, en commençant aux coordonnées *src_x*, *src_y* et sur la largeur de *src_w* et la hauteur de *src_h*. La portion ainsi définie sera copiée et placée aux coordonnées *dst_x* et *dst_y*.

10.32.17 ImageCopyResized

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecopyresized](#)(int *dst_im*, int *src_im*, int *dstX*, int *dstY*, int *srcX*, int *srcY*, int *dstW*, int *dstH*, int *srcW*, int *srcH*)

[imagecopyresized\(\)](#) copie une partie rectangulaire d'une image dans une autre image de destination. *dst_im* est l'image de destination, *src_im* est l'image source. Si les dimensions de la source et de la destination ne sont pas égales, un étirement adéquat est effectué pour faire correspondre les deux. Les coordonnées fournies se repère par rapport au coin supérieur gauche. Cete fonction peut être utilisée pour recopier des régions à l'intérieur d'une même image, si *dst_im* et *src_im* sont identiques : mais si les régions se chevauchent, le résultat risque d'être incohérent.

10.32.18 ImageCreate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecreate](#)(int *x_size*, int *y_size*)

[imagecreate\(\)](#) retourne un identifiant d'image représentant une image blanche, de largeur *x_size* et longueur *y_size*.

10.32.19 ImageCreateFromGif

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecreatefromgif](#)(string *filename*)

[imagecreatefromgif\(\)](#) retourne un identifiant d'image qui représente l'image obtenue à partir du fichier dont le nom est donné.

[imagecreatefromgif\(\)](#) retourne une chaîne vide en cas d'échec. Il va aussi retourner une erreur qui va afficher un lien brisé dans un navigateur. Pour simplifier le débogage, utilisez le code suivant, qui retourne une erreur GIF : **Exemple de gestion des erreurs durant création d'image (gracieusement offert par courtesy vic@zysys.com)**

```
function LoadGif($imgname)
{
    $im = @imagecreatefromgif($imgname); /* Tentative d'ouverture */
    if ($im == "") { /* Echec ? */
        $im = ImageCreate(150,30); /* Crée une image vide */
        $bgc = ImageColorAllocate($im,255,255,255);
        $tc = ImageColorAllocate($im,0,0,0);
        ImageFilledRectangle($im,0,0,150,30,$bgc);
        ImageString($im,1,5,5,"Erreur lors du chargement du fichier $imgname",$tc); /* Affiche un message */
    }
    return $im;
}
```

Note : *Etant donné que toutes les fonctions de gestion des GIF ont été supprimées de la bibliothèque GD version 1.6, cette fonction n'est pas disponible si vous utilisez cette version de la librairie.*

10.32.20 ImageCreateFromJPEG

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecreatefromjpeg](#)(string *filename*)

[imagecreatefromjpeg\(\)](#) retourne un identifiant d'image représentant un image obtenu à partir du fichier *filename*.

[imagecreatefromjpeg\(\)](#) retourne une chaîne vide en cas d'échec. Elle affiche aussi un message d'erreur, qui s'affiche comme un lien brisé dans un navigateur web. Pour faciliter le débogage, voici une erreur **JPEG**: exemple de gestion d'erreur lors de la création d'image (gracieusement offert par vic@zysmsys.com) *exemple de gestion d'erreur lors de la création d'image (gracieusement offert par vic@zysmsys.com)*

```

function LoadJpeg ($imgname) {
    $im = @ImageCreateFromJPEG ($imgname); /* Tentative d'ouverture */
    if (!$im) { /* Vérification */
        $im = ImageCreate (150, 30); /* Création d'une image blanche */
        $bgc = ImageColorAllocate ($im, 255, 255, 255);
        $tc = ImageColorAllocate ($im, 0, 0, 0);
        ImageFilledRectangle ($im, 0, 0, 150, 30, $bgc);
        /* Affichage d'un message d'erreur */
        ImageString ($im, 1, 5, 5, "Erreur de chargement de l'image $imgname", $tc);
    }
    return $im;
}

```

10.32.21 ImageCreateFromPNG

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecreatefrompng](#)(string *filename*)

[imagecreatefrompng\(\)](#) retourne un identifiant d'image représentant un image obtenu à partir du fichier *filename*.

[imagecreatefrompng\(\)](#) retourne une chaîne vide en cas d'échec. Elle affiche aussi un message d'erreur, qui s'affiche comme un lien brisé dans un navigateur web. Pour faciliter le débogage, voici une erreur **PNG**: exemple de gestion d'erreur lors de la création d'image (gracieusement offert par vic@zysmsys.com) *exemple de gestion d'erreur lors de la création d'image (gracieusement offert par vic@zysmsys.com)*

```

function LoadPNG ($imgname) {
    $im = @ImageCreateFromPNG ($imgname); /* Tentative d'ouverture */
    if (!$im) { /* Vérification */
        $im = ImageCreate (150, 30); /* Création d'une image blanche */
        $bgc = ImageColorAllocate ($im, 255, 255, 255);
        $tc = ImageColorAllocate ($im, 0, 0, 0);
        ImageFilledRectangle ($im, 0, 0, 150, 30, $bgc);
        /* Affichage d'un message d'erreur */
        ImageString ($im, 1, 5, 5, "Erreur de chargement de l'image $imgname", $tc);
    }
    return $im;
}

```

}

10.32.22 ImageDashedLine

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagedashedline](#) (int *im*, int *x1*, int *y1*, int *x2*, int *y2*, int *col*)

[imagedashedline\(\)](#) dessine une ligne pointillée entre les points (x1,y1) et (x2,y2) (le coin supérieur droit est l'origine (0,0)) dans l'image *im*, avec la couleur *col*.

Voir aussi [imageline\(\)](#).

10.32.23 ImageDestroy

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagedestroy](#) (int *im*)

[imagedestroy\(\)](#) libère toute la mémoire associée avec l'image *im*. *im* est un identifiant d'image valide retourné par [imagecreate\(\)](#).

10.32.24 ImageFill

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagefill](#) (int *im*, int *x*, int *y*, int *col*)

[imagefill\(\)](#) effectue un remplissage avec la couleur *col*, dans l'image *im*, à partir du point de coordonnées (x,y) (le coin supérieur gauche est l'origine (0,0)).

10.32.25 ImageFilledPolygon

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagefilledpolygon](#) (int *im*, array *points*, int *num_points*, int *col*)

[imagefilledpolygon\(\)](#) dessine un polygone rempli dans l'image *im*. *points* est un tableau PHP qui contient les sommets des polygones sous la forme :. points[0] = x0, points[1] = y0, points[2] = x1, points[3] = y1, etc. *num_points* est le nombre total de sommets.

10.32.26 ImageFilledRectangle

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagefilledrectangle](#) (int *im*, int *x1*, int *y1*, int *x2*, int *y2*, int *col*)

[imagefilledrectangle\(\)](#) dessine un rectangle de couleur *col* dans l'image *im*, en commençant par le sommet supérieur gauche (*x1*, *y1*) et finissant au sommet inférieur droit (*x2*, *y2*). Le coin supérieur gauche est l'origine (0, 0).

10.32.27 ImageFillToBorder

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagefilltoborder\(\)](#) (int *im*, int *x*, int *y*, int *border*, int *col*)

[imagefilltoborder\(\)](#) remplit avec la couleur *col* toute la région à l'intérieur de la région limitée par la couleur *border*. Le point de départ est (*x,y*) (le coin supérieur gauche est l'origine (0,0)).

10.32.28 ImageFontHeight

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagefontheight\(\)](#) (int *font*)

[imagefontheight\(\)](#) retourne la hauteur de la police en pixel.
Voir aussi [imagefontwidth\(\)](#) et [imageloadfont\(\)](#).

10.32.29 ImageFontWidth

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagefontwidth\(\)](#) (int *font*)

[imagefontwidth\(\)](#) retourne la largeur de la police en pixels.
Voir aussi [imagefontheight\(\)](#) et [imageloadfont\(\)](#).

10.32.30 ImageGif

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagegif\(\)](#) (int *im*, string *filename*)

[imagegif\(\)](#) crée un fichier image GIF avec le nom *filename* d'après l'image *im*. L'argument *im* est un identifiant valide retourné par la fonction [imagecreate\(\)](#).

Le format de l'image sera GIF87a à moins que l'image n'ait une couleur transparente (mise en place grâce à la fonction [imagecolortransparent\(\)](#)), ce qui fera qu'elle sera au format GIF89a.

Le nom du fichier est optionnel, et dans ce cas, l'image sera transmise directement à la sortie standard. En envoyant une image de type image/gif content-type, (grâce à la fonction [header\(\)](#)), vous pouvez créer des images avec des scripts PHP. Note : *Etant donné que toutes les fonctions GIF ont été supprimées de la bibliothèque GD version 1.6, cette fonction ne sera pas accessible si vous avez cette version de la librairie. Le code suivant vous permet d'écrire des scripts PHP plus portables : le type de GD est automatiquement détecté. Il remplace la séquence Header("Content-type: image/gif"); ImageGif(\$im); par un code plus souple :*

```
<?php
if (function_exists("imagegif")) {
Header("Content-type: image/gif");
ImageGif($im);

elseif (function_exists("imageJpeg")) {
Header("Content-type: image/jpeg");
ImageJpeg($im, "", 0.5);
```

```

    }
elseif (function_exists("imagePng")) {
Header("Content-type: image/png");
ImagePng($im);
}
else
die("Pas de support graphique avec PHP sur ce serveur");
?>

```

} Note : En PHP4, à partir de la version 4.0.2, vous pouvez utiliser la fonction [imagetypes\(\)](#) à la place de [function_exists\(\)](#) pour vérifier que certains formats d'images sont supportés :

```

if (ImageTypes() & IMG_GIF) {
Header("Content-type: image/gif");
ImageGif($im);

elseif (ImageTypes() & IMG_JPG) {
... etc.

```

}
Voir aussi [imagepng\(\)](#), [imagejpeg\(\)](#), [imagetypes\(\)](#).

10.32.31 ImagePNG

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagepng\(\)](#) (int *im*, string *filename*)

[imagepng\(\)](#) envoie l'image GD (*im*) au format PNG sur la sortie standard (typiquement, le navigateur web), ou si *filename* est fourni, l'envoi dans un fichier.

```

<?php
$im = ImageCreateFromPng("test.png");
ImagePng($im);
?>

```

Le nom du fichier est optionnel, et dans ce cas, l'image sera transmise directement à la sortie standard. En envoyant une image de type image/png content-type, (grâce à la fonction [header\(\)](#)), vous pouvez créer des images PNG avec des scripts PHP.

Voir aussi [imagegif\(\)](#), [imagejpeg\(\)](#), [imagetypes\(\)](#).

10.32.32 ImageJPEG

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagejpeg\(\)](#) (int *im*, string *filename* , int *quality*)

[imagejpeg\(\)](#) envoie l'image GD (*im*) au format JPEG sur la sortie standard (typiquement, le navigateur web), ou si *filename* est fourni, l'envoi dans un fichier. *im* a été créé par [imagecreate\(\)](#).

Le nom du fichier est optionnel, et dans ce cas, l'image sera transmise directement à la sortie standard. En envoyant une image de type image/jpeg content-type, (grâce à la fonction [header\(\)](#)), vous pouvez créer des images JPEG avec des scripts PHP.

Note : Le support JPEG n'est disponible que si PHP est compilé avec GD-1.8 ou plus récent.
Voir aussi [imagepng\(\)](#), [imagegif\(\)](#), [imagetypes\(\)](#).

[10.32.33 ImageInterlace](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imageinterlace](#)(int *im*, int *interlace*)

[imageinterlace\(\)](#) active ou désactive le bit d'entrelacement. Si l'entrelacement est à 1, l'image *im* sera interlacée, et sinon, elle ne le sera pas.

Cette fonction retourne l'état courant d'entrelacement de l'image.

[10.32.34 ImageLine](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imageline](#)(int *im*, int *x1*, int *y1*, int *x2*, int *y2*, int *col*)

[imageline\(\)](#) dessine une ligne depuis le point (*x1,y1*) jusqu'au point (*x2,y2*) (le coin supérieur gauche est l'origine (0,0)) dans l'image *im* et avec la couleur *col*.

Voir aussi [imagecreate\(\)](#) et [imagecolorallocate\(\)](#).

[10.32.35 ImageLoadFont](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imageloadfont](#)(string *file*)

[imageloadfont\(\)](#) charge une nouvelle police utilisateur et retourne un identifiant sur cette police. Cet identifiant sera toujours supérieur à 5, pour éviter les conflits avec les polices standard PHP).

Le format des polices dépend actuellement du système d'exploitation. Ce qui signifie qu'il vous faut générer des fichiers de polices pour la machine qui fait tourner PHP.

position	Type de données C	description
Octets 0-3	int	Nombre de caractères de la police
Octets 4-7	int	Valeur du premier caractère de la police (souvent 32 pour espace) @tab
Octets 8-11	int	Largeur en pixel des caractères
Octets 12-15	int	Hauteur en pixel des caractères
Octets 16-	char	Tableau avec les données des caractères, un octet par pixel pour chaque caractère, avec un total de (nombre_caractères*largeur*hauteur) octets. @tab

Voir aussi [imagefontwidth\(\)](#) et [imagefontheight\(\)](#).

10.32.36 ImagePsExtendFont

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [imagepsextendfont](#) (int *font_index* , double *extend*)

Etend ou condense la police de caractères *font_index*. Si la valeur de *extend* est inférieure à 1, ce sera une condensation.

10.32.37 ImagePsSlantFont

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [imagepslantfont](#) (int *font_index* , double *slant*)

Met en italique la police de caractères *font_index* avec le coefficient *slant*.

10.32.38 ImagePolygon

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagepolygon](#) (int *im*, array *points*, int *num_points*, int *col*)

[imagepolygon\(\)](#) dessine un polygone dans l'image *im*. *points* est un tableau PHP qui contient les sommets du polygone sous la forme : points[0] = x0, points[1] = y0, points[2] = x1, points[3] = y1, etc. num_points est le nombre de sommets.

Voir aussi [imagecreate\(\)](#).

10.32.39 ImagePSBBox

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imagepsbbox](#) (string *text*, int *font*, int *size*, int *space*, int *width*, float *angle*)

size est exprimé en pixels.

space permet de changer la valeur par défaut du caractère espace. Cette valeur est ajoutée lors des dessins, et donc, peut être négative.

tightness permet de contrôler la quantité d'espace entre les caractères. Cette quantité est ajoutée lors des dessins, et peut donc être négative.

angle est en degrés.

Les paramètres *space* et *tightness* sont exprimés en unité d'espacement de caractères, avec 1 unité vaut 1/1000 d'un em carré (? ? ?).

Les paramètres *space*, *tightness* et *angle* sont optionnels.

Le rectangle entourant est calculé en utilisant les informations disponibles sur les tableaux de caractères, et, malheureusement, ont tendance à être légèrement différent du résultat réel final. Si l'angle est de 0 degré, vous pouvez vous attendre à avoir besoin d'un rectangle d'au moins un pixel plus grand dans toutes les directions.

Cette fonction retourne un tableau contenant les éléments suivants :

0	Abscisse inférieure gauche
1	Ordonnée inférieure gauche

2	Abscisse supérieure droite
3	Ordonnée supérieure droite

Voir aussi [imagepstext\(\)](#).

[10.32.40 ImagePSEncodeFont](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagepsencodefont](#) (string *encodingfile*)

[imagepsencodefont\(\)](#) charge le codage vectoriel d'un caractère depuis un fichier et change le codage vectoriel de la police correspondante. Etant donné que les polices PostScript ne disposent pas des caractères au-delà de 127, vous aurez sûrement besoin de les changer sur vous utilisez une autre langue que l'anglais. Le format exact est décrit dans la documentation T1libs. T1lib est disponible en deux formes : IsoLatin1.enc et IsoLatin2.enc.

Si vous commencez à utiliser cette fonction régulièrement, une meilleure solution est de définir un encodage, et de l'utiliser avec `set ps.default_encoding` dans [7.1 Le fichier de configuration](#) pour utiliser par défaut l'encodage correct.

[10.32.41 ImagePSFreeFont](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [imagepsfreefont](#) (int *fontindex*)

Voir aussi [imagepsloadfont\(\)](#).

[10.32.42 ImagePSLoadFont](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagepsloadfont](#) (string *filename*)

Au cas où tout a bien marché, un index de police va être retourné, et pourra être utilisé pour des opérations ultérieures. Sinon, la fonction retourne FALSE et affiche un message décrivant ce qui est erroné.

Voir aussi [imagepsfreefont\(\)](#).

[10.32.43 ImagePSText](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imagepstext](#) (int *image*, string *text*, int *font*, int *size*, int *foreground*, int *background*, int *x*, int *y*, int *space*, int *tightness*, float *angle*, int *antialias_steps*)

size est exprimé en pixels.

foreground est la couleur dans laquelle le texte va être dessiné. *background* est la couleur d'anti aliasing.

Aucun pixel avec la couleur *background* n'est dessiné, ce qui fait que l'arrière plan n'a pas besoin d'être dans une couleur fixe.

Les coordonnées données (*x*, *y*) définissent l'origine du premier caractère (grossièrement, le coin inférieur

gauche du caractère). Ceci est différent de la fonction [imagestring\(\)](#), où (x, y) définissait le coin supérieur gauche du premier caractère. Reportez vous à la documentation PostScript pour avoir des détails à propos des polices et de leurs tailles.

space permet de changer la taille par défaut du caractère d'espacement. Cette valeur peut être négative.

tightness permet de contrôler la quantité d'espace entre deux caractères. Cette valeur peut être négative.

angle est en degrés.

antialias_steps permet de contrôler le nombre de couleurs du texte anti-aliasé. Les valeurs autorisées sont 4 et 16. 16 est recommandé pour les polices de moins de 20 pixels, car l'effet est alors visible. Avec les tailles plus grandes, utilisez de préférence 4, qui est moins gourmande en ressources.

Les paramètres *space* et *tightness* sont exprimés en unité d'espace caractère, ce qui vaut 1/1000ème d'un em-carré (? ? ?).

Les paramètres *space*, *tightness*, *angle* et *antialias* sont optionnels.

Cette fonction retourne un tableau contenant les éléments suivants :

0	Abscisse inférieure gauche
1	Ordonnée inférieure gauche
2	Abscisse supérieure droite
3	Ordonnée supérieure droite

Voir aussi [imagepsbbox\(\)](#).

[10.32.44 ImageRectangle](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagerectangle](#) (int *im*, int *x1*, int *y1*, int *x2*, int *y2*, int *col*)

[imagerectangle\(\)](#) dessine un rectangle dans la couleur *col*, dans l'image *im*, et en commençant au point supérieur gauche (*x1,y1*), et en finissant au point inférieur droit (*x2,y2*). Le coin supérieur gauche est l'origine (0,0).

[10.32.45 ImageSetPixel](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagesetpixel](#) (int *im*, int *x*, int *y*, int *col*)

[imagesetpixel\(\)](#) dessine un pixel au point (x,y) (le coin supérieur gauche est l'origine (0,0)) dans l'image *im*, et avec la couleur *col*.

Voir aussi [imagecreate\(\)](#) et [imagecolorallocate\(\)](#).

[10.32.46 ImageString](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagestring](#) (int *im*, int *font*, int *x*, int *y*, string *s*, int *col*)

[imagestring\(\)](#) dessine une ligne horizontale, dans l'image *im*, aux coordonnées (x,y) (le coin supérieur gauche est l'origine (0,0)) dans la couleur *col*. Si l'argument de police vaut 1, 2, 3, 4 ou 5, une des polices par défaut sera utilisée).

Voir aussi [imagerloadfont\(\)](#).

[10.32.47 ImageStringUp](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagestringup](#) (int *im*, int *font*, int *x*, int *y*, string *s*, int *col*)

[imagestringup\(\)](#) dessine une chaîne verticale dans l'image *im* aux coordonnées (*x*, *y*) (l'origine est le coin supérieur gauche (0,0)) dans la couleur *col*. Si la police utilisée est 1, 2, 3, 4 ou 5, une police par défaut sera utilisée.

Voir aussi [imagerloadfont\(\)](#).

[10.32.48 ImageSX](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagesx](#) (int *im*)

[imagesx\(\)](#) retourne la largeur de l'image référencée par *im*.

Voir aussi [imagecreate\(\)](#) et [imagesy\(\)](#).

[10.32.49 ImageSY](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagesy](#) (int *im*)

[imagesy\(\)](#) retourne la hauteur de l'image référencée par *im*.

Voir aussi [imagecreate\(\)](#) et [imagesx\(\)](#).

[10.32.50 ImageTTFBBox](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imaggettffbbox](#) (int *size*, int *angle*, string *fontfile*, string *text*)

[imaggettffbbox\(\)](#) calcule et retourne le rectangle entourant le texte *text*, écrit avec une police truetype.

text

- La chaîne à mesurer.
size
- La taille de la police.
fontfile
- Le nom de la police TrueType (peut aussi être une URL.)
angle
- Angle en degré dans lequel le texte *text* va être mesuré.

[imaggettffbbox\(\)](#) retourne une tableau avec 8 éléments, représentant les 4 sommets du rectangle ainsi définis.

0	Coin inférieur gauche, abscisse
1	Coin inférieur gauche, ordonnée
2	Coin inférieur droit, abscisse
3	Coin inférieur droit, ordonnée
4	Coin supérieur droit, abscisse
5	Coin supérieur droit, ordonnée
6	Coin supérieur gauche, abscisse
7	Coin supérieur gauche, ordonnée

Les positions des points sont relatives au texte *text*, indépendamment de l'angle : coin supérieur gauche faire référence au coin supérieur gauche du texte écrit horizontalement.

Cette fonction requiert les bibliothèques GD et Freetype.

Voir aussi [imaggotbbox\(\)](#).

[10.32.51 ImageTTFText](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imaggotbbox\(\)](#) (int *im*, int *size*, int *angle*, int *x*, int *y*, int *col*, string *fontfile*, string *text*)

[imaggotbbox\(\)](#) dessine la chaîne *text* dans l'image *im*, en commençant aux coordonnées (*x,y*) (le coin supérieur gauche est l'origine (0,0)), avec un angle de *angle*, et dans la couleur *col*, en utilisant la police TrueType identifiée par *fontfile*.

Les coordonnées (*x,y*) serviront de référence pour le premier caractère (en gros, le coin inférieur gauche du caractère). C'est différent de [imagestring\(\)](#), qui utilise le coin supérieur droit.

angle est donné en degrés, avec degré 0 pour un texte horizontal, et en comptant les angles dans le sens inverse des aiguilles d'une montre (sens direct).

fontfile est le chemin jusqu'à la police TrueType à utiliser.

text est le texte à dessiner, incluant aussi des séquences de caractères UTF-8 (de la forme: `{`) pour générer des caractères au delà de 255.

col est l'index de la couleur dans la palette. Utiliser des index négatifs, revient à supprimer l'anti-aliasing.

[imaggotbbox\(\)](#) retourne un tableau de 8 éléments représentant les 4 points marquant les limites du texte. L'ordre des points est :supérieur gauche, supérieur droit, inférieur droit, inférieur gauche. Les points sont nommés relativement au texte à l'horizontal.

Cet exemple va générer une image GIF noire de 400x30 pixels, avec les mots "Test en cours..." en police blanche, Arial. *ImageTTFText*

```
<?php
Header("Content-type: image/gif");
$im = imagecreate(400,30);
$black = ImageColorAllocate($im, 0,0,0);
$white = ImageColorAllocate($im, 255,255,255);
ImageTTFText($im, 20, 0, 10, 20, $white, "/path/arial.ttf", "Test en cours... Omega: &#937;");
ImageGif($im);
ImageDestroy($im);
?>
```

Cette fonction requiert les bibliothèques GD et [FreeType](#).
Voir aussi [imageftbbox\(\)](#).

[10.32.52 ImageTypes](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imageTypes](#)

Cette fonction retourne un champs de bit correspondant aux formats d'images supportés par la version de GD utilisée. Les valeurs suivantes sont valables : IMG_GIF | IMG_JPG | IMG_PNG | IMG_WBMP. Pour vous assurer du support PNG, faite ceci : *Exemple avec ImageTypes*

```
<?php
if (ImageTypes() & IMG_PNG) {
echo "Le type PNG est supporté";
}
?>
```

[10.32.53 read_exif_data](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [read_exif_data](#) (string *filename*)

[read_exif_data\(\)](#) lis les entêtes EXIF de l'image JPEG *filename*. Elle retourne un tableau associatif où les index sont les noms d'entêtes EXIF, et les valeurs sont leur valeur associée. Les entêtes EXIF sont souvent disponibles dans les images générées par les appareils phots digitaux, mais chaque constructeur marque ses images d'une manière qui lui est propre : il est impossible de savoir quels entêtes seront présents.

read_exif_data

```
<?php
$exif = read_exif_data ('p0001807.jpg');
while(list($k,$v)=each($exif)) {
echo "$k: $v<br>\n";
}
?>
```

Output:

```
FileName: p0001807.jpg
FileDateTime: 929353056
FileSize: 378599
CameraMake: Eastman Kodak Company
CameraModel: KODAK DC265 ZOOM DIGITAL CAMERA (V01.00)
DateTime: 1999:06:14 01:37:36
Height: 1024
Width: 1536
IsColor: 1
FlashUsed: 0
FocalLength: 8.0mm
```

```
RawFocalLength: 8
ExposureTime: 0.004 s (1/250)
RawExposureTime: 0.0040000001899898
ApertureFNumber: f/ 9.5
RawApertureFNumber: 9.5100002288818
FocusDistance: 16.66m
RawFocusDistance: 16.659999847412
Orientation: 1
ExifVersion: 0200
```

Note : Cette fonction n'est disponible que sous PHP 4, compilé avec `--enable-exif`. Cette fonction ne requiert par la librairie GD.

10.33 Fonctions IMAP

[\[Notes en ligne\]](#)

Pour avoir accès à ces fonctions, vous devez compiler PHP avec l'option `--with-imap`. Il faut avoir installé la librairie `c-client`. Chargez sa dernière version sur le serveur <ftp://ftp.cac.washington.edu/imap/> et compilez la. Puis, copiez le fichier ``c-client/c-client.a'` dans ``usr/local/lib'` ou n'importe quel autre dossier qui soit dans le chemin de link. Enfin, copiez les fichiers ``c-client/rfc822.h'`, ``mail.h'` et ``linkage.h'` dans ``usr/local/include'` ou n'importe quel autre dossier qui soit dans le chemin d'inclusion.

Ces fonctions ne sont pas limitées au protocole **IMAP**, malgré leur nom. La librairie sur laquelle elles sont développées supporte aussi **NNTP**, **POP3** et les méthodes d'accès aux boîtes aux lettres locales. Reportez vous à la fonction [imap_open\(\)](#) pour plus d'informations.

10.33.1 imap_append

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_append\(\)](#)(int *imap_stream*, string *mbx*, string *message*, string *flags*)

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[imap_append\(\)](#) ajoute un message dans la boîte aux lettres *mbx*. Si l'option *flags* est utilisée, *flags* sera aussi écrit dans la boîte aux lettres.

Lors des échanges avec le serveur Cyrus IMAP, vous devrez utiliser `"\r\n"` comme terminaison de ligne, à la place de `"\n"` ou l'opération échouera.

10.33.2 imap_base64

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_base64\(\)](#)(string *text*)

[imap_base64\(\)](#) décode un texte encodé en BASE64. Le texte décodé est retourné sous la forme d'une chaîne.

10.33.3 imap_body

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_body](#) (int *imap_stream*, int *msg_number*, int *flags*)

[imap_body\(\)](#) retourne le corps du message numéro *msg_number* de la boîte aux lettres courante. L'option *flags* est un masque qui peut contenir les valeurs suivantes :

- FT_UID – msgno est un UID
- FT_PEEK – Ne pas lever le drapeaux \Seen (Message lu) si il n'est pas déjà levé.
- FT_INTERNAL – La chaîne renvoyée est au format interne, et ne va pas canoniser les CRLF.

[10.33.4 imap_check](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [imap_check](#) (int *imap_stream*)

[imap_check\(\)](#) retourne les informations à propos de la boîte aux lettres courante. Retourne FALSE en cas d'échec.

La fonction [imap_check\(\)](#) vérifie le statut de la boîte aux lettres courante, sur le serveur *imap_stream*, et retourne les informations dans un objet avec les membres suivants :

```
Date : : Date du message
Driver : Pilote
Mailbox : Nom de la boîte aux lettres
Nmsgs : Nombre de messages
Recent : Nombre de messages récents
```

[10.33.5 imap_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_close](#) (int *imap_stream*, int *flags*)

Termine un flot IMAP. Prend un argument optionnel *flag*, CL_EXPUNGE, qui va retirer automatiquement de la liste la boîte aux lettres.

[10.33.6 imap_createmailbox](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_createmailbox](#) (int *imap_stream*, string *mbox*)

[imap_createmailbox\(\)](#) crée une nouvelle boîte aux lettres nommée *mbox* (voir [imap_open\(\)](#) pour connaître le format des noms de *mbox*).

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

Voir aussi [imap_renamemailbox\(\)](#) et [imap_deletemailbox\(\)](#).

[10.33.7 imap_delete](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_delete](#) (int *imap_stream*, int *msg_number*)

Returns TRUE.

[imap_delete\(\)](#) marque le fichier *msg_number* pour l'effacement, dans la boîte aux lettres courante. L'effacement réel n'interviendra que lors de l'appel de la fonction [imap_expunge\(\)](#).

[10.33.8 imap_deletemailbox](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_deletemailbox](#) (int *imap_stream*, string *mbx*)

[imap_deletemailbox\(\)](#) efface la boîte aux lettres (voir [imap_open\(\)](#) pour connaître le format des noms de *mbx*).

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

Voir aussi [imap_createmailbox\(\)](#) et [imap_renamemailbox\(\)](#).

[10.33.9 imap_expunge](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_expunge](#) (int *imap_stream*)

[imap_expunge\(\)](#) efface tous les messages marqués pour l'effacement par [imap_delete\(\)](#).

Retourne TRUE.

[10.33.10 imap_fetchbody](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_fetchbody](#) (int *imap_stream*, int *msg_number*, string *part_number*, flags *flags*)

Cette fonction va rechercher une section du corps du message, et la retourne sous la forme d'une chaîne. La section est une chaîne d'entiers, séparés par des virgules, qui servent d'index dans le corps du message, comme spécifié dans la norme IMAP4. Le texte n'est alors pas décodé par [imap_fetchbody\(\)](#).

L'option [imap_fetchbody\(\)](#) est un masque qui peut contenir les valeurs suivantes :

- FT_UID – msgono est un UID
- FT_PEEK – Ne pas lever le drapeau \Seen (Message lu) si il n'est pas déjà levé.
- FT_INTERNAL – La chaîne renvoyée est au format interne, et ne va pas canoniser les CRLF.

10.33.11 imap_fetchstructure

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [imap_fetchstructure](#) (int *imap_stream*, int *msg_number*, int *flags*)

Lit la structure du message *msg_number*. Cette fonction dispose d'une option *flags*, qui une seule valeur, *FT_UID*, pour indiquer que l'argument *msg_number* est un *UID*. Cette fonction retourne un objet avec des propriétés d'enveloppe, de date interne, de taille, de structure de flags et de corps, ainsi qu'un objet pour chaque attachement. La structure est la suivante :

type	Type primaire de corps
encoding	Codage de transfert du corps
ifsubtype	TRUE si il y a une chaîne de sous type
subtype	sous type <i>MIME</i>
ifdescription	TRUE si il y au ne chaîne de description
description	Chaîne de description du contenu
ifid	TRUE si il y a une chaîne d'identification
id	Chaîne d'identification
lines	Nombre de lignes
bytes	Nombre d'octets
ifdisposition	TRUE si il y a une chaîne de disposition
disposition	Chaîne de disposition
ifdparameters	TRUE s'il y a un tableau de paramètres dparameters
dparameters dparameters est un tableau d'objet où chaque objet à un "attribut" et une "valeur". @tab tableau de disposition	
ifparameters	TRUE si le tableau de paramètres existe
paramètres parameter est un tableau d'objet où chaque objet à un "attribut" et une "valeur". @tab Tableau de paramètres <i>MIME</i>	
parts parts est un tableau d'objets de même structure que l'objet supérieur, mais qui ne contient pas d'autres objets de même sorte. @tab Tableau d'objet décrivant chaque partie du message	

0	text
1	multipart
2	message
3	application
4	audio
5	image
6	video
7	other

0	7BIT
1	8BIT
2	BINARY
3	BASE64
4	QUOTED-PRINTABLE
5	OTHER

[10.33.12 imap_header](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [imap_header](#) (int *imap_stream*, int *msg_number*, int *fromlength* , int *subjectlength* , string *default host*)

Cette fonction retourne un objet avec divers éléments d'entête :

```
remail, date, Date, subject, Subject, in_reply_to, message_id,
newsgroups, followup_to, references
```

message flags:

```
Recent - 'R' si récent et lu
          'N' si récent et pas lu
          ' ' if par récent
Unseen - 'U' si pas lu ET pas récent
          ' ' si pas lu OU pas lu et récent
Answered - 'A' si répondu
           ' ' si non répondu
Deleted - 'D' si effacé
           ' ' si pas effacé
Draft - 'X' si brouillon
         ' ' si pas brouillon
Flagged - 'F' si marqué
          ' ' si non marqué
```

NOTE : le comportement de Récent/pas lu est un peu étrange. Si vous voulez savoir si un message n'est pas lu, vous devez vérifier avec
Unseen == 'U' || Recent == 'N'

```

toaddress (ligne to: complète, jusqu'à 1024 caractères)
to[] (retourne un tableau d'objets, extrait de la ligne To , et contenant)
personal
adl
mailbox
host
fromaddress (ligne From: complète, jusqu'à 1024 caractères)
from[] (retourne un tableau d'objets, extrait de la ligne From , et contenant)
personal
adl
mailbox
host
ccaddress (ligne cc: complète, jusqu'à 1024 caractères)
cc[] (retourne un tableau d'objets, extrait de la ligne Cc , et contenant)
personal
adl
mailbox
host
bccaddress (ligne bcc: complète, jusqu'à 1024 caractères)
bcc[] (retourne un tableau d'objets, extrait de la ligne Bcc , et contenant)
personal
adl
mailbox
host
reply_toaddress (ligne reply_to: complète, jusqu'à 1024 caractères)
reply_to[] (retourne un tableau d'objets, extrait de la ligne Reply_to , et contenant)
personal
adl
mailbox
host
senderaddress (ligne sender_address: complète, jusqu'à 1024 caractères)
sender[] (retourne un tableau d'objets, extrait de la ligne Reply_to , et contenant)
personal
adl
mailbox
host
return_path (ligne return_path: complète, jusqu'à 1024 caractères)
return_path[] (retourne un tableau d'objets, extrait de la ligne return_path , et contenant)
personal
adl
mailbox
host
update (date du message, au format unix (timestamps))
fetchfrom (Ligne From , limitée à fromlength caractères)
fetchsubject (Ligne de sujet, limitée à subjectlength caractères)

```

[10.33.13 imap_headers](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_headers](#) (int *imap_stream*)

[imap_headers\(\)](#) retourne un tableau de chaîne contenant les entête des messages. Une chaîne par message.

[10.33.14 imap_listmailbox](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_listmailbox](#) (int *imap_stream*, string *ref*, string *pat*)

Retourne un tableau contenant les noms des boîtes aux lettres.

[10.33.15 imap_getmailboxes](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_getmailboxes](#) (int *imap_stream*, string *ref*, string *pat*)

Retourne un tableau d'objets contenant les informations sur les boîtes aux lettres. Chaque objet a les attributs de *name*, qui contient le nom complet de la boîte aux lettres; *delimiter*, qui est le délimiteur hiérarchique; et *attributes*. *Attributes* est un masque de bits, qui contient :

- LATT_NOINFERIORS – Cette boîte aux lettres n'a pas d'"enfants" (il n'y a plus de boîtes aux lettres en dessous de celle-ci).
- LATT_NOSELECT – Ceci est juste un container, pas une boîte aux lettres (vous ne pouvez pas l'ouvrir).
- LATT_MARKED – Cette boîte aux lettres est marquée. Utilisé uniquement avec UW-IMAPD.
- LATT_UNMARKED – Cette boîte aux lettres n'est pas marquée. Utilisé uniquement avec UW-IMAPD.

ref ne devrait être que le serveur IMAP sous la forme {imap_server:imap_port}, et *pattern* spécifie la position dans la hiérarchie des boîtes aux lettres, où il faut commencer à chercher. Si vous voulez passer en revue toute la hiérarchie, passez '*' comme *pattern*.

Il y a deux caractères spéciaux que vous pouvez utiliser dans *pattern* : '*' et '%'. '*' signifie : toutes les boîtes aux lettres. Si vous passez *pattern* comme '*', vous obtiendrez la liste complète des boîtes aux lettres de la hiérarchie. '%' signifie qu'on ne s'intéresse qu'au niveau courant. '%' passé à *pattern* ne retournera que les boîtes aux lettres de niveau supérieur; '~/mail/%'. Sous UW-IMAPD retournera toutes les boîtes aux lettres du dossier '~/mail directory', mais pas leurs enfants.

[10.33.16 imap_listsubscribed](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_listsubscribed](#) (int *imap_stream*, string *ref*, string *pattern*)

[imap_listsubscribed\(\)](#) retourne un tableau avec toutes les boîtes aux lettres auxquelles vous avez souscrit. Les arguments *ref* et *pattern* indiquent respectivement, le dossier où chercher et le nom des boîtes recherchées, sous la forme d'un masque.

[10.33.17 imap_getsubscribed](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_getsubscribed](#) (int *imap_stream*, string *ref*, string *pattern*)

Cette fonction est identique à [imap_getmailboxes\(\)](#), mais ne retourne que les boîtes aux lettres auxquelles l'utilisateur est inscrit.

[10.33.18 imap_mail_copy](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_mail_copy](#)(int *imap_stream*, string *msglist*, string *mbox*, int *flags*)

[imap_mail_copy\(\)](#) retourne TRUE en cas de succès et FALSE en cas d'erreur.

[imap_mail_copy\(\)](#) copie les messages email spécifiés par *msglist* dans la boîte aux lettres nommée *mbox*. *msglist* est un intervalle, et pas seulement une liste numéros de message.

flags est un masque, qui peut contenir une ou plusieurs des valeurs suivantes :

- CP_UID – la séquence de nombre contient des UIDS
- CP_MOVE – Efface les messages après copie.

[10.33.19 imap_mail_move](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_mail_move](#)(int *imap_stream*, string *msglist*, string *mbox*)

[imap_mail_move\(\)](#) déplace les messages spécifiés par *msglist* dans la boîte aux lettres *mbox*. *msglist* est un intervalle, et pas seulement une liste de messages.

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[10.33.20 imap_num_msg](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_num_msg](#)(int *imap_stream*)

[imap_num_msg\(\)](#) retourne le nombre de message dans la boîte aux lettres courante.

[10.33.21 imap_num_recent](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_num_recent](#)(int *imap_stream*)

[imap_num_recent\(\)](#) retourne le nombre de message récents dans la boîte aux lettres courante.

[10.33.22 imap_open](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_open](#)(string *mailbox*, string *username*, string *password*, int *flags*)

Retourne un flot IMAP en cas de succès, et FALSE en cas d'erreur. Cette fonction peut aussi être utilisée

pour ouvrir des flots sur des serveurs POP3 et NNTP. Pour se connecter à un serveur IMAP, on peut utiliser la commande suivante :

```
$mbox = imap_open( "{localhost:143}INBOX", "user_id", "password" );
```

Pour se connecter à un serveur POP3 qui fonctionne sur le port 110 de la machine locale on peut utiliser la commande suivante :

```
$mbox = imap_open( "{localhost/pop3:110}INBOX", "user_id", "password" );
```

Pour se connecter à un serveur NNTP qui fonctionne sur le port 119 de la machine locale on peut utiliser la commande:

```
$nntp = imap_open( "{localhost/nntp:119}comp.test", "", "" );
```

Pour se connecter à un serveur distant, remplacez "localhost" par le nom ou l'adresse IP de la machine. Les options sont un masque de bit, qui peut prendre une ou plusieurs des valeurs suivantes :

- OP_READONLY – Ouvre une boîte aux lettres en lecture seule
- OP_ANONYMOUS – Ne pas utiliser, ou modifier le fichier .newsrc pour les news.
- OP_HALFOPEN – Pour les noms IMAP et NNTP, ouvre une connexion mais n'ouvre pas une boîte aux lettres.
- CL_EXPUNGE – Supprime automatiquement la boîte aux lettres de la liste, lors de la terminaison du flot.

[10.33.23 imap_ping](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

```
int imap\_ping(int imap_stream)
```

Retourne TRUE si le flot existe toujours, et FALSE sinon.

[imap_ping\(\)](#) vérifie que le flot IMAP est toujours actif, en lui envoyant un ping. Cette fonction permet de se rendre compte que du mail est arrivé : c'est même la méthode préconisée pour des tests périodiques de vérification du courrier. Cette fonction peut aussi servir à garder une connexion ouverte, avec les serveurs dotés d'un délai d'expiration.

[10.33.24 imap_renamemailbox](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

```
int imap\_renamemailbox(int imap_stream, string old_mbox, string new_mbox)
```

Renomme une boîte aux lettres.

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

Voir aussi [imap_createmailbox\(\)](#) et [imap_deletemailbox\(\)](#).

[10.33.25 imap_reopen](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_reopen](#) (string *imap_stream*, string *mailbox*, string *flags*)

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

Cette fonction ré-ouvre le flot spécifié, mais vers une nouvelle boîtes aux lettres.

Les options sont des masques de bit, qui peuvent contenir les valeurs suivantes :

- OP_READONLY – Ouvre une boîte aux lettres en lecture seule
- OP_ANONYMOUS – Ne pas utiliser, ou modifier le fichier .newsrc pour les news
- OP_HALFOPEN – Pour les noms IMAP et NNTP, ouvre une connexion mais n'ouvre pas une boîte aux lettre.
- CL_EXPUNGE – Supprime automatiquement la boîte aux lettres de la liste, lors de la terminaison du flot.

[10.33.26 imap_subscribe](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_subscribe](#) (int *imap_stream*, string *mbox*)

Souscrit à la boîte aux lettres *mbox*.

Retourne TRUE en cas de succès, et FALSE en cas d'échec.

[10.33.27 imap_undelete](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_undelete](#) (int *imap_stream*, int *msg_number*)

Enlève la marque d'effacement du message *msg_number*, placée avec [imap_delete\(\)](#).

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[10.33.28 imap_unsubscribe](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_unsubscribe](#) (int *imap_stream*, string *mbox*)

Termine la souscription à la boîte aux lettres *mbox*.

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[10.33.29 imap_qprint](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_qprint](#) (string *string*)

Convertit la chaîne à guillemets *string* en une chaîne à 8 bits.
Retourne une chaîne 8 bits (binaire).

[10.33.30 imap_8bit](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_8bit](#) (string *string*)

Convertit la chaîne à 8 bits en une chaîne à guillemets.
Retourne une chaîne à guillemets.

[10.33.31 imap_binary](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_binary](#) (string *string*)

Convertit la chaîne à 8 bits *string* en une chaîne à base64.
Retourne la chaîne codée.

[10.33.32 imap_scanmailbox](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_scanmailbox](#) (int *imap_stream*, string *string*)

Lis la liste des boîtes aux lettres, et y recherche la chaîne *string*.

[10.33.33 imap_mailboxmsginfo](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [imap_mailboxmsginfo](#) (int *imap_stream*)

Retourne les informations à propos de la boîte aux lettres courante. Retourne FALSE en cas d'échec.
[imap_mailboxmsginfo\(\)](#) vérifie le statut courant de la boîte aux lettres sur le serveur, et retourne un objet avec les propriétés suivantes :

```
Date : date du message
Driver : lecteur
Mailbox : Nom de la boîte aux lettres
Nmsgs : nombre de messages
Recent : nombre de messages récents
Unread : nombre de messages non lus
Size : taille de la boîte aux lettres
```

[10.33.34 imap_rfc822_write_address](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_rfc822_write_address](#)(string *mailbox*, string *host*, string *personal*)

Retourne une adresse email proprement formatée, à partir du nom de la boîte aux lettres de l'hôte, et des informations personnelles.

[10.33.35 imap_rfc822_parse_adrlist](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_rfc822_parse_adrlist](#)(string *address*, string *default_host*)

Cette fonction analyse la chaîne *address* et essaie, pour chaque adresse, retourner un tableau d'objets. les 4 objets sont :

```
mailbox - Le nom de la boîte aux lettres (nom d'utilisateur)
host - Le nom de l'hôte
personal - Le nom de l'utilisateur
adl - La route jusqu'au domaine
```

[10.33.36 imap_setflag_full](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_setflag_full](#)(int *stream*, string *sequence*, string *flag*, string *options*)

Cette fonction affecte le flag spécifié aux messages de la séquence donnée.

Les flags que vous pouvez modifier sont "\\Seen", "\\Answered", "\\Flagged", "\\Deleted", "\\Draft" et "\\Recent" (comme défini dans la RFC2060).

Les options sont un masque de bits, et peuvent contenir les valeurs suivantes :

```
ST_UID la séquence contient des UIDs au lieu de numéro de séquence
```

[10.33.37 imap_clearflag_full](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_clearflag_full](#)(int *stream*, string *sequence*, string *flag*, string *options*)

Cette fonction efface le flag spécifié dans les messages de la séquence *sequence*.

Les options sont un masque de bit, qui accepte les valeurs suivantes :

```
ST_UID la séquence contient des UIDs au lieu de numéro de séquence
```

10.33.38 imap_sort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_sort](#) (int *stream*, int *criteria*, int *reverse*, int *options*)

Retourne un tableau de nombre de message, triés suivant les paramètres suivants :

Rev vaut 1 pour signifier : tri inverse (REVERSE-sorting).

Les critères peuvent être un (et un seul) parmi les suivants :

SORTDATE	Date du message
SORTARRIVAL	Date d'arrivée
SORTFROM	Nom de la première boîte aux lettres de l'adresse d'origine (From address)
SORTSUBJECT	Sujet du message
SORTTO	Nom de la première boîte aux lettres de destination (To address)
SORTCC	Nom de la boîte aux lettres de copie cachée (cc address)
SORTSIZE	Taille du message en octets

Les flags dont des masques de bits, d'un ou plusieurs des éléments suivants :

SE_UID	Retourne l'UIDs à la place d'une séquence de nombres
SE_NOPREFETCH	Ne pas pré-télécharger les messages trouvés

10.33.39 imap_fetchheader

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_fetchheader](#) (int *imap_stream*, int *msgno*, int *flags*)

Cette fonction retourne l'entête brut et complet RFC 822 du message *msgno*, et le retourne sous la forme d'une chaîne.

Les options sont :

FT_UID	L'argument msgno est un UID
FT_INTERNAL	la chaîne renvoyée est au format "internal" , c'est à dire sans canonisation des CRLF
FT_PREFETCHTEXT	RFC822.TEXT doit être pré-téléchargé en même temps que l'entête. Cela réduit le R

10.33.40 imap_uid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_uid](#) (int *imap_stream*, int *msgno*)

Cette fonction retourne l'UID pour le message *msgno*. C'est la fonction contraire de [imap_msgno\(\)](#).

10.33.41 imap_msgno

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_msgno](#)(int *imap_stream*, int *uid*)

[imap_msgno\(\)](#) retourne le numéro de séquence de message pour l'UID *uid*. C'est la fonction contraire de [imap_uid\(\)](#).

10.33.42 imap_search

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_search](#)(int *imap_stream*, string *criteria*, int *flags*)

[imap_search\(\)](#) effectue une recherche dans la boîte aux lettres courantes, sur le flot IMAP courant. *criteria* est une chaîne, délimitée par des espaces, dans laquelle les mots-clés suivants sont acceptés : Tous les arguments multi-mots doivent être entre guillemets :

- ALL – retourne tous les message qui vérifie le reste du critère.
- ANSWERED – tous les messages avec le flag `\\ANSWERED`
- BCC "string" – tous les messages avec la chaîne "string" dans le champs Bcc:
- BEFORE "date" – tous les messages avec Date: avant "date"
- BODY "string" – tous les messages avec "string" dans le corps
- CC "string" – tous les messages avec "string" dans le champs Cc:
- DELETED – tous les messages effacés
- FLAGGED – tous les messages avec le flag `\\FLAGGED` (parfois interprété comme Important ou Urgent)
- FROM "string" – tous les messages avec la chaîne "string" dan le champs From:
- KEYWORD "string" – tous les messages avec la chaîne "string" comme mot clé
- NEW – tous les nouveaux messages
- OLD – tous les anciens messages
- ON "date" – tous les messages avec la date "date" comme champs Date:
- RECENT – tous les messages avec le flag `\\RECENT`
- SEEN – tous les messages lus (avec le flag `\\SEEN` flag)
- SINCE "date" – tous les messages avec la date Date: après "date"
- SUBJECT "string" – tous les messages avec la chaîne "string" dans le champs Subject:
- TEXT "string" – tous les messages avec le texte "string"
- TO "string" – tous les messages avec la chaîne "string" dans le champs To:
- UNANSWERED – tous les messages non répondu
- UNDELETED – tous les messages non effacés
- UNFLAGGED – tous les messages non flaggés
- UNKEYWORD "string" – tous les messages dans le mot clés "string"
- UNSEEN – tous les messages non lus

Par exemple, pour rechercher les messages non répondu, envoyés par maman, vous pouvez utiliser : "UNANSWERED FROM maman". Les recherches semblent insensibles à la casse. Cette liste de critères est issue du code d'un client C UW et peut être incomplète ou imprécise. Faites attention.

Les valeurs pour les flags sont SE_UID, qui fait que le tableau réponse contient les UIDs plutôt que les numéros de séquence.

10.33.43 imap_last_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_last_error](#)(void)

Cette fonction retourne le texte complet de la dernière erreur IMAP (si elle existe) qui est survenu lors de la dernière requête. La pile d'erreur n'est pas touchée. Appeler [imap_last_error\(\)](#) successivement dans nouvelles erreurs retournera la même erreur.

ATTENTION : cette fonction n'est pas encore disponible sous PHP4.

10.33.44 imap_errors

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_errors](#)(void)

Cette fonction retourne tous les messages d'erreurs IMAP générés depuis le dernier appel à [imap_errors\(\)](#), ou depuis le début de la page. Lorsque [imap_errors\(\)](#) est appelés, la pile d'erreur est vidée.

ATTENTION : cette fonction n'est pas encore disponible sous PHP4.

10.33.45 imap_alerts

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_alerts](#)(void)

Cette fonction retourne tous les messages d'alerte IMAP générés depuis le dernier appel à [imap_alerts\(\)](#) ou depuis le début de la page. Lorsque [imap_alerts\(\)](#) est appelé, la pile d'alerte est vidée.

10.33.46 imap_status

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [imap_status](#)(int *imap_stream*, string *mailbox*, int *options*)

Cette fonction retourne un objet contenant les informations de statut. Les flags valables sont :

- SA_MESSAGES – met la valeur de status->messages au nombre de messages dans la boîtes aux lettres.
- SA_RECENT – met la valeur destatus->recent au nombre de messages récents dans la boîte aux lettres.
- SA_UNSEEN – met la valeur de status->unseen au nombre de messages non lus dans la boîte aux lettres.
- SA_UIDNEXT – met la valeur de status->uidnext à la prochaine valeur d'uid qui sera utilisée.
- SA_UIDVALIDITY – met la valeur de status->uidvalidity à une constante, qui change lorsque l'uid de la boîte aux lettres n'est plus valide.
- SA_ALL – fixe les valeurs de de toutes les précédents.

status->flags est aussi fixé : c'est un masque de bit qui peut contenir tous les flags ci dessus.

[10.33.47 imap_utf7_decode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_utf7_decode](#) (string *text*)

Décode la chaîne UTF-7 *text* en données 8 bits.

Retourne les données décodées en 8 bits, ou FALSE si la chaîne n'était pas une chaîne UTF-7 valide.

[10.33.48 imap_utf7_encode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_utf7_encode](#) (string *data*)

Convertit les données 8 bits *data* en texte UTF-7 *text*. L'encodage UTF-7 est défini dans la RFC 2060.

Retourne un texte UTF-7.

[10.34 options PHP & informations](#)

[\[Notes en ligne\]](#)

[10.34.1 extension_loaded](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [extension_loaded](#) (string *name*)

Retourne vraie si l'extension *name* a été chargée. Vous pouvez voir les différents noms des extensions, en utilisant la fonction [phpinfo\(\)](#).

Voir aussi [phpinfo\(\)](#). Note : *Cette fonction a été ajoutée dans 3.0.10.*

[10.34.2 getenv](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [getenv](#) (string *varname*)

Retourne la valeur de la variable d'environnement *varname*, ou FALSE en cas d'erreur.

```
$ip = getenv("REMOTE_ADDR"); // retourne l'adresse IP de l'utilisateur
```

Vous pouvez voir une liste complète des variables d'environnement en utilisant la fonction [phpinfo\(\)](#). Vous pouvez trouver la signification de chacune d'entre elles en consultant le site concernant [CGI specification](#) (en anglais>, et particulièrement la page concernant les [variables d'environnement](#).

[10.34.3 get_cfg_var](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [get_cfg_var](#)(string *varname*)

Retourne la valeur courante de l'option PHP *varname*, ou bien FALSE en cas d'erreur.

Cette fonction ne retourne pas les options qui ont été choisies lors de la compilation de PHP, ni ne lit dans le fichier de configuration d'Apache.

Pour vérifier si le système utilise le [7.1 Le fichier de configuration](#), essayez de lire la valeur de `cfg_file_path`. Si cette valeur est disponible, alors le fichier de configuration est utilisé.

[10.34.4 get_current_user](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [get_current_user](#)(void)

Retourne le nom du possesseur du script courant.

Voir aussi [getmyuid\(\)](#), [getmypid\(\)](#), [getmyinode\(\)](#), et [getlastmod\(\)](#).

[10.34.5 get_magic_quotes_gpc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

long [get_magic_quotes_gpc](#)(void)

Retourne la configuration actuelle de l'option [7.1.1.16 ini.magic-quotes-gpc](#) (0 pour l'option désactivée, 1 pour l'option activée).

Voir aussi [get_magic_quotes_runtime\(\)](#), [set_magic_quotes_runtime\(\)](#).

[10.34.6 get_magic_quotes_runtime](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

long [get_magic_quotes_runtime](#)(void)

Retourne la configuration actuelle de l'option [7.1.1.17 ini.magic-quotes-runtime](#). (0 pour option désactivée, 1 pour option activée).

Voir aussi [get_magic_quotes_gpc\(\)](#), [set_magic_quotes_runtime\(\)](#).

[10.34.7 getlastmod](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getlastmod](#)(void)

Retourne la date de dernière modification de la page. La valeur retournée est un marqueur de temps UNIX, utilisable comme paramètre avec la fonction [date\(\)](#). Retourne FALSE en cas d'erreur. *getlastmod()* **example**

```
// affiche 'Dernière modification: March 04 1998 20:43:59.'
echo "Dernière modification: ".date( "F d Y H:i:s.", getlastmod() );
```

Voir aussi [date\(\)](#), [getmyuid\(\)](#), [get_current_user\(\)](#), [getmyinode\(\)](#), et [getmypid\(\)](#).

10.34.8 getmyinode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getmyinode](#)(void)

Retourne l'inode du script, ou FALSE en cas d'erreur.

Voir aussi [getmyuid\(\)](#), [get_current_user\(\)](#), [getmypid\(\)](#), et [getlastmod\(\)](#).

Note : *Cette fonction est inopérante sur les systèmes Windows.*

10.34.9 getmypid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getmypid](#)(void)

Retourne le numéro de processus actuel ou FALSE en cas d'erreur.

Il est à noter que si vous utilisez PHP comme module Apache, il n'est pas garanti que deux invocations distinctes de la fonction donnent des résultats différents.

Voir aussi [getmyuid\(\)](#), [get_current_user\(\)](#), [getmyinode\(\)](#), et [getlastmod\(\)](#).

10.34.10 getmyuid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getmyuid](#)(void)

Retourne l'UID du propriétaire du script actuel ou FALSE en cas d'erreur.

Voir aussi [getmypid\(\)](#), [get_current_user\(\)](#), [getmyinode\(\)](#), et [getlastmod\(\)](#).

10.34.11 getrusage

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [getrusage](#)(int *who*)

Cette fonction est une interface à la fonction system `getrusage(2)`. Elle retourne un tableau associatif contenant les informations renvoyées par cet appel système. Si "who is 1", `getrusage` sera appelé avec le paramètre `RUSAGE_CHILDREN`.

Toutes les valeurs du tableau sont accessibles en utilisant leur nom dans le tableau. *Exemple `getrusage`*

```
$dat = getrusage();
echo $dat["ru_nswap"];           # Taille de la mémoire swap
echo $dat["ru_majflt"];         # Nombre de page mémoires utilisées
echo $dat["ru_utime.tv_sec"];   # temps utilisateur (en secondes)
echo $dat["ru_utime.tv_usec"]; # temps utilisateur (en microsecondes)
```

Consultez le manuel "man" pour plus de détails.

[10.34.12 phpinfo](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [phpinfo](#)(void)

Affiche de nombreuses informations sur le PHP, concernant sa configuration courante : options de compilation, extensions, version, informations sur le serveur, et environnement (lorsque compilé comme module), environnement PHP, chemins, utilisateur, entêtes HTTP, et licence GNU Public License. Voir aussi [phpversion\(\)](#).

[10.34.13 phpversion](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [phpversion](#)(void)

Retourne le numéro de la version courante de PHP. *exemple de la fonction [phpversion\(\)](#)*

```
// affiche le numéro de version courante du PHP.  
echo "PHP Version: ".phpversion();
```

Voir aussi [phpinfo\(\)](#).

[10.34.14 php logo guid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [php_logo_guid](#)(void)

Note : *Cette fonctionnalité a été ajoutée dans PHP4 Beta 4.*

[10.34.15 putenv](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [putenv](#)(string *setting*)

Fixe la valeur d'une variable d'environnement.

Fixer la valeur d'une variable d'environnement

```
putenv("UNIQID=$uniqid");
```

[10.34.16 set_magic_quotes_runtime](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

long [set_magic_quotes_runtime](#)(int *new_setting*)

Active/désactive l'option [7.1.1.17 ini.magic-quotes-runtime](#). (0 l'option est désactivée, 1 l'option est activée). Voir aussi [get_magic_quotes_gpc\(\)](#) et [get_magic_quotes_runtime\(\)](#).

[10.34.17 set_time_limit](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [set_time_limit](#)(int *seconds*)

Fixe le délai d'expiration d'un script, en secondes. Si cette limite est atteinte, le script s'interrompt, et renvoie une erreur fatale. La valeur par défaut est 30 secondes ou, si c'est le cas, la valeur de la directive `max_execution_time` définie dans le [7.1 Le fichier de configuration](#). Si la valeur est zéro, il n'y a alors aucune limite imposée.

Lorsqu'elle est appelée, la fonction [set_time_limit\(\)](#) remet le compteur de zéro. En d'autres termes, si la limite par défaut est à 30 secondes, et qu'après 25 secondes d'exécution du script l'appel `set_time_limit(20)` est fait, alors le script tournera pendant un total de 45 secondes avant de finir.

Notez que [set_time_limit\(\)](#) n'a pas d'effet lorsque PHP fonctionne en mode "safe mode". Il n'y a pas d'autre solution que de changer de mode, ou de modifier la durée maximale d'exécution dans le [7.1 Le fichier de configuration](#).

[10.34.18 zend_logo_guid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [zend_logo_guid](#)(void)

Note : *Cette fonctionnalité a été ajouté en PHP4 Beta 4.*

[10.35 Fonctions LDAP](#)

[\[Notes en ligne\]](#)

[10.35.1 Introduction à LDAP](#)

[\[Notes en ligne\]](#)

LDAP est l'acronyme de Lightweight Directory Access Protocol, c'est à dire Protocole Léger d'Accès aux Dossiers. C'est un protocole utilisé pour accéder à des "serveurs de dossiers", des serveurs qui gèrent les informations de manière hiérarchique.

Le concept est similaire à la structure de votre disque dur, hormis le fait que la racine s'appelle ici : "The world" (le monde), et que les dossiers du premier niveau sont assimilés à des pays. Les niveaux inférieurs de la structure contiennent des entrées de sociétés, d'organisations ou de lieux tandis que les niveaux encore inférieurs sont des gens, voire des équipement ou des documents.

Pour accéder à un fichier sur votre disque, vous devez utiliser la syntaxe suivante :

```
/usr/local/myapp/docs
```

Le slash indique une division de la référence, et la séquence est lue de gauche à droite. Une telle référence en LDAP sera exprimée avec une autre syntaxe, la syntaxe à "noms distincts" ("distinguished names"), aussi appelé "nd" ("dn" en anglais). Par exemple :

```
cn=Jean Dupont,ou=Comptes,o=Ma Société,c=Fr
```

La virgule marque une division de la référence, et la séquence est lue de droite à gauche. Vous pouvez la lire comme ceci :

```
country = Fr
organization = Ma Société
organizationalUnit = Comptes
commonName = Jean Dupont
```

De la même façon qu'il n'y a pas de règle universelle d'organisation d'un disque dur, un serveur de dossier peut supporter n'importe quelle structure du moment qu'elle a un sens pour ce qu'on en fait. Cependant, il existe quelques conventions : il est impossible d'écrire un code d'accès à un dossier sans en connaître sa structure, de la même façon que vous ne pouvez pas utiliser une base de données sans en connaître les tables.

10.35.2 Exemple complet

[\[Notes en ligne\]](#)

Recupérer toutes les entrées dont le nom commence par "S" dans un serveur, et afficher le nom et l'adresse email.

Recherche LDAP

```
// Structure d'une commande simple :
// connexion, lien, recherche, interpretation de la recherche
// résultat, déconnexion
<?php
echo "<?h3>LDAP query test</h3>";
echo "Connexion ...";
$ds=ldap_connect("localhost"); // Doit être un serveur LDAP valide!
echo "Résultat de la connexion : ".$ds."<?p>";
if ($ds) {
echo "Lien ...";
    $r=ldap_bind($ds); // Ceci est un lien "anonymous", typiquement
                      // en lecture seule. En cas d'accès, affiche
                      // " Lien résultat est"
echo "Lien résultat est ".$r."<?p>";
echo "Recherche de (sn=S*) ...";
    // Recherche dans les noms
    $sr=ldap_search($ds,"o=Ma Société, c=Fr", "sn=S*");
echo "Résultat : ".$sr."<?p>";
echo "Nombre d'entrée retournée : ".ldap_count_entries($ds,$sr)."<?p>";
echo "Lecture des entrées...<?p>";
    $info = ldap_get_entries($ds, $sr);
echo "Data for ".$info["count"]." items returned:<?p>";
for ($i=0; $i<?$info["count"]; $i++) {
echo "dn vaut : ". $info[$i]["dn"] ."<?br>";
echo "première entrée cn vaut : ". $info[$i]["cn"][0] ."<?br>";
echo "première email vaut: ". $info[$i]["mail"][0] ."<?p>";
}
```

```

    }
    echo "Déconnexion ";
    ldap_close($ds);
  } else {
    echo "<?h4>Impossible de se connecter à un serveur LDAP </h4>";
  }
?>

```

[10.35.2.1 Utilisation des fonctions PHP LDAP](#)

[\[Notes en ligne\]](#)

Il faut d'abord que les bibliothèques client LDAP soient compilées avec PHP. Vous pouvez vous procurer ces bibliothèques University of Michigan (ldap-3.3 package) ou chez Netscape (Netscape Directory SDK). Avant d'utiliser les fonctions LDAP il faut savoir :

- Le nom ou l'adresse du serveur à utiliser
- Le "nd" dans le serveur (la partie du monde qui est sur ce serveur, ce qui peut correspondre à "o=Ma société,c=Fr")
- Eventuellement, un mot de passe pour accéder au serveur (de nombreux serveur fournissent un accès en lien anonymes ("anonymous bind") mais requièrent un mot de passe pour tout le reste).

Une séquence habituelle LDAP suivra le schéma suivant :

```

ldap_connect()    // établi une connexion à un serveur
|
ldap_bind()      // nom de compte "login" ou anonyme
|
  exécution de commandes sur le serveur, comme un listage, ou
  une modification de données avec affichage
|
ldap_close()     // "déconnexion"

```

[10.35.2.2 Plus d'informations](#)

[\[Notes en ligne\]](#)

Vous pouvez en apprendre encore plus, mais en anglais, aux adresses suivantes :

- [Netscape](#)
- [University of Michigan](#)
- [OpenLDAP Project](#)
- [LDAP World](#)

Le SDK Netscape contient un guide du programmeur au format HTML particulièrement pratique.

[10.35.3 ldap_add](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_add](#)(int *link_identifiant*, string *dn*, array *entry*)

Retourne TRUE en cas de succès, ou FALSE en cas d'erreur.

[ldap_add\(\)](#) sert à ajouter une entrée dans un dossier LDAP. Le ND de l'entrée sera ajouté à la **dn** du dossier spécifié. Le tableau **entry** spécifie les informations de la nouvelle entrée. Les valeurs de l'entrée sont indexées dans les attributs de l'entrée. Si un attribut a de multiples valeurs, elles seront indexées dans un tableau, à partir de l'index 0.

```
entree["attribut1"] = valeur
entree["attribut2"][0] = valeur1
entree["attribut2"][1] = valeur2
```

Exemple complet avec lien authentifié

```
<?php
$ds=ldap_connect("localhost"); // On suppose que le serveur LDAP est sur cet hote
if ($ds) {
    // liaison avec le nd approprié, pour avoir un accès en modification
    $r=ldap_bind($ds,"cn=root, o=Ma Société, c=Fr", "secret");
    // preparation des données
    $info["cn"]="John Jones";
    $info["sn"]="Jones";
    $info["mail"]="jonj@here.and.now";
    $info["objectclass"]="person";
    // Ajout des données dans le dossier
    $r=ldap_add($ds, "cn=John Jones, o=My Company, c=US", $info);
    ldap_close($ds);
} else {
    echo "Impossible de se connecter au serveur LDAP ";
}
?>
```

10.35.4 ldap_mod_add

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_mod_add](#)(int *link_identifieur*, string *dn*, array *entry*)

Retourne TRUE en cas de succès, et FALSE sinon.

Cette fonction ajoute les attributs dans la nd spécifié. Cet ajout est fait au niveau attribut, et non pas au niveau objet. Les ajouts au niveau objet sont faits par la fonction [ldap_add\(\)](#).

10.35.5 ldap_mod_del

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_mod_del](#)(int *link_identifieur*, string *dn*, array *entry*)

Retourne TRUE en cas de succès, et FALSE sinon.

Cette fonction efface les attributs du nd spécifié. Cette modification est fait au niveau des attributs, et par opposition au niveau de l'objet. Les effacements au niveau objet sont effectués par la fonction [ldap_delete\(\)](#).

10.35.6 ldap_mod_replace

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_mod_replace](#) (int *link_identifieur*, string *dn*, array *entry*)

Retourne TRUE en cas de succès, et FALSE sinon.

Cette fonction remplace les attributs dans du nd *dn* spécifié. Cette modification est faite au niveau attribut, et non pas au niveau objet. Les modifications au niveau objet sont faites par la fonction [ldap_modify\(\)](#).

[10.35.7 ldap_bind](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_bind](#) (int *link_identifieur*, string *bind_rdn*, string *bind_password*)

[ldap_bind\(\)](#) lie un serveur LDAP avec le RDN et mot de passe spécifié (éventuellement). Retourne TRUE en cas de succès, et FALSE sinon.

[ldap_bind\(\)](#) effectue une opération de liaison sur le serveur. *bind_rdn* et *bind_password* sont optionnels. Si ils manquent, la liaison se fera en mode anonyme.

[10.35.8 ldap_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_close](#) (int *link_identifieur*)

Retourne TRUE en cas de succès, et FALSE sinon.

[ldap_close\(\)](#) ferme le lien au serveur LDAP associé à l'identifiant *link_identifieur*.

Cet appel est identique à [ldap_unbind\(\)](#), en interne. Les API LDAP utilisent la fonction [ldap_unbind\(\)](#): il est probablement mieux que vous utilisiez cette fonction là plutôt que [ldap_close\(\)](#).

[10.35.9 ldap_connect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_connect](#) (string *hostname*, int *port*)

Retourne un identifiant positif de serveur LDAP en cas de succès, ou bien FALSE en cas d'erreur.

[ldap_connect\(\)](#) établit une connexion avec un serveur. Le serveur LDAP situé sur l'hôte *hostname* et *port*.

Les deux arguments sont optionnels. Sans argument, l'identifiant de la dernière connexion ouverte sera retournée. Si seul *hostname* est spécifié, le port par défaut est 389.

[10.35.10 ldap_count_entries](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_count_entries](#) (int *link_identifieur*, int *result_identifieur*)

Retourne le nombre d'entrées en cas de succès, et FALSE sinon.

[ldap_count_entries\(\)](#) retourne le nombre d'entrée placées dans le résultat par les recherches précédentes. *result_identifieur* identifie un résultat LDAP interne.

[10.35.11 ldap_delete](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_delete](#)(int *link_identifieur*, string *dn*)

Retourne TRUE en cas de succès, et FALSE sinon.

[ldap_delete\(\)](#) efface une entrée dans un dossier LDAP spécifié par le nd *dn*.

[10.35.12 ldap_dn2ufn](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ldap_dn2ufn](#)(string *dn*)

[ldap_dn2ufn\(\)](#) sert à mettre le nd *dn* dans un format plus agréable, notamment en supprimant les noms des types.

[10.35.13 ldap_explode_dn](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ldap_explode_dn](#)(string *dn*, int *with_attrib*)

[ldap_explode_dn\(\)](#) sert à scinder le nd *dn* retourné par [ldap_get_dn\(\)](#) en plusieurs composants. Chaque composant est reconnu sous le nom Nom Distinct Relatif (ou RDN, en anglais). [ldap_explode_dn\(\)](#) retourne un tableau qui contient ces composants. *with_attrib* sert à préciser si le RDN est retourné avec ses attributs, ou seul. Pour obtenir le RDN et ses attributs, mettez *with_attrib* à 0 et pour n'avoir que les valeurs, mettez le à 1.

[10.35.14 ldap_first_attribute](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ldap_first_attribute](#)(int *link_identifieur*, int *result_entry_identifieur*, int *ber_identifieur*)

Retourne le premier attribut en cas de succès, et FALSE sinon.

Le comportement est similaire pour les entrées. Les attributs sont lus séquentiellement dans une entrée particulière. [ldap_first_attribute\(\)](#) retourne le premier attribut de l'entrée désignée par l'identifiant d'entrée.

Les attributs suivants sont accessibles avec [ldap_next_attribute\(\)](#). *ber_identifieur* est un identifiant de pointeur de mémoire interne. Il est passé par référence. Le même identifiant *ber_identifieur* est passé à

[ldap_next_attribute\(\)](#), qui modifie ce pointeur.

Voir aussi [ldap_get_attributes\(\)](#).

[10.35.15 ldap_first_entry](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_first_entry](#)(int *link_identifieur*, int *result_identifieur*)

Retourne un identifiant sur la première entrée en cas de succès, et FALSE sinon.

Les entrées d'un résultat sont lues séquentiellement, en utilisant [ldap_first_entry\(\)](#) et [ldap_next_entry\(\)](#). [ldap_first_entry\(\)](#) retourne l'identifiant de la première entrée du résultat. Cet identifiant sera fourni à [ldap_next_entry\(\)](#) pour accéder à la prochaine entrée.

Voir aussi [ldap_get_entries\(\)](#).

[10.35.16 ldap_free_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_free_result](#) (int *result_identifieur*)

Retourne TRUE en cas de succès, et FALSE sinon.

[ldap_free_result\(\)](#) libère la mémoire allouée en interne pour enregistrer le résultat pointé par *result_identifieur*.

A la fin de chaque script, la mémoire sera de toute manière libérée.

Généralement, il n'y a pas besoin de libérer la mémoire, et le mécanisme automatique de fin de script est suffisant. Cependant, dans les cas où le script effectue plusieurs recherches successives, où que les résultats retournés sont très grands, [ldap_free_result\(\)](#) permet de réduire la consommation de mémoire.

[10.35.17 ldap_get_attributes](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ldap_get_attributes](#) (int *link_identifieur*, int *result_entry_identifieur*)

Retourne un tableau multi-dimensionnel en cas de succès, et FALSE sinon.

[ldap_get_attributes\(\)](#) sert à simplifier la lecture des attributs et des valeurs d'une entrée dans un résultat. Le résultat est un tableau multi-dimensionnel, avec les attributs en clé, et les valeurs des attributs en valeurs.

Une fois que vous avez repéré une entrée dans un dossier, vous pouvez lire les informations de cette entrée avec cette fonction. Vous pouvez utiliser cette fonction pour créer une application qui se déplace dans les dossiers, sans en connaître la structure au préalable. Dans de nombreux cas, vous ne cherchez qu'un attribut particulier (le email, par exemple) et vous ne vous intéressez pas aux autres valeurs.

Affichage de la liste des attributs d'une entrée

```
// $ds est l'identifiant de lien pour ce dossier
// $sr est un résultat de recherche valide, obtenu lors d'une recherche
// précédente
$entry = ldap_first_entry($ds, $sr);
$sattrs = ldap_get_attributes($ds, $entry);
echo $sattrs["count"]." Attributs dans cette entrée:<p>";
for ($i=0; $i<$sattrs["compte"]; $i++)
echo $sattrs[$i]."<br>";
```

Voir aussi [ldap_first_attribute\(\)](#) et [ldap_next_attribute\(\)](#).

[10.35.18 ldap_get_dn](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ldap_get_dn](#) (int *link_identifieur*, int *result_entry_identifieur*)

Retourne le DN de l'entrée en cas de succès, et FALSE sinon.
[ldap_get_dn\(\)](#) sert à obtenir le ND d'une entrée d'un résultat.

10.35.19 ldap_get_entries

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ldap_get_entries](#)(int *link_identifiant*, int *result_identifiant*)

Retourne un tableau multi dimensionnel en cas de succès, et FALSE sinon.

[ldap_get_entries\(\)](#) sert à simplifier la lecture d'un résultat à plusieurs entrées. Toutes les informations sont retournées sous la forme d'un tableau multi-dimensionnel. La structure de ce tableau est la suivante : Les attributs servent d'index et sont mis en minuscule (Les attributs sont insensibles à la casse sur les serveurs, mais peuvent ne pas l'être quand ils sont utilisés comme index)

```

résultat ["compte"] = nombre d'entrées du résultat
résultat [0] : correspond aux détails de la première entrée :
résultat [i]["nd"] = ND de la i-ième entrée
résultat [i]["compte"] = nombre d'attributs de la i-ième entrée
résultat [i][j] = j-ième attribut de la i-ième entrée
résultat [i]["attribut"]["count"] = nombre de valeur pour l'attribut
résultat [i]["attribut"][j] = j-ième valeur de l'attribut

```

Voir aussi [ldap_first_entry\(\)](#) et [ldap_next_entry\(\)](#).

10.35.20 ldap_get_values

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ldap_get_values](#)(int *link_identifiant*, int *result_entry_identifiant*, string *attribute*)

Retourne un tableau de valeurs en cas de succès, et FALSE sinon.

[ldap_get_values\(\)](#) sert à lire toutes les valeurs d'un attribut dans une entrée. L'entrée est référencée par *result_entry_identifiant*. Le nombre de valeurs peut être trouvé à l'index "count" dans le résultat. Les valeurs sont accessibles par un index entier, qui commence à 0.

Cette fonction nécessite un pointeur de résultat *result_entry_identifiant*, ce qui implique qu'il ai été précédé d'une recherche sur le serveur, et de l'obtention d'une entrée.

Votre application pourra utiliser des noms d'attributs en dur dans le code, ou bien, utiliser la fonction [ldap_get_attributes\(\)](#) pour y accéder dynamiquement.

LDAP autorise plus d'une entrée par attribut, ce qui permet, par exemple, d'étiqueter tous les adresses email d'un utilisateur avec l'attribut "mail"

```

return_value["count"] = nombre de valeurs de l'attribut
return_value[0] = première valeur de l'attribut
return_value[i] = n-ième valeur de attribut

```

Liste toutes les valeurs avec l'attribut "mail"

```

// $ds est l'identifiant de lien pour ce dossier
// $sr est un résultat de recherche valide, obtenu lors d'une recherche
// précédente

```

```
// $entry est un identifiant valide d'entrée
$values = ldap_get_values($ds, $entry, "mail");
echo $values["count"]." Adresse email dans ce résultat.<p>";
for ($i=0; $i < $values["count"]; $i++)
echo $values[$i]."<br>";
```

[10.35.21 ldap_get_values_len](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ldap_get_values_len](#)(int *link_identifieur*, int *result_entry_identifieur*, string *attribut*)

Retourne un tableau de valeurs pour l'attribut, ou bien FALSE en cas d'erreur.

[ldap_get_values_len\(\)](#) sert à lire toutes les valeurs d'un attribut d'une entrée dans un résultat. L'entrée est spécifiée par *result_entry_identifieur*. Le nombre de valeurs trouvées peut être retrouvé en indexant "count" dans le tableau résultat. On peut accéder aux valeurs individuelles avec un index numérique, commençant à 0. Cette fonction est utilisée exactement comme [ldap_get_values\(\)](#) mais elle gère les données binaires, et non pas les chaînes.

[10.35.22 ldap_list](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_list](#)(int *link_identifieur*, string *base_dn*, string *filter*, array *attributes*)

Retourne TRUE en cas de succès, et FALSE sinon.

[ldap_list\(\)](#) effectue une recherche avec le filtre donnée, et limité un dossier.

LDAP_SCOPE_ONELEVEL indique que la recherche ne doit s'étendre que dans le dossier immédiatement sous le nd *base_dn*. (Equivalent à taper "ls" et obtenir la liste des fichiers et dossiers du dossier courant).

Cette appel prend un quatrième argument optionnel : un tableau contenant les attributs recherchés. Reportez vous à [ldap_search\(\)](#) pour plus de détails. *Affiche une liste d'unités organisationnelle*

```
// $ds est un identifiant de connexion valide.
$basedn = "o=Ma Société, c=Fr";
$justthese = array("ou");
$sr=ldap_list($ds, $basedn, "ou=*", $justthese);
$info = ldap_get_entries($ds, $sr);
for ($i=0; $i<$info["count"]; $i++)
echo $info[$i]["ou"][0] ;
```

[10.35.23 ldap_modify](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_modify](#)(int *link_identifieur*, string *dn*, array *entry*)

Retourne TRUE en cas de succès, et FALSE sinon.

[ldap_modify\(\)](#) sert à modifier les entrées existantes dans un dossier LDAP. La structure de l'entrée est la même que décrite dans [ldap_add\(\)](#).

[10.35.24 ldap_next_attribute](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ldap_next_attribute](#) (int *link_identifiant*, int *result_entry_identifiant*, int *ber_identifiant*)

Retourne l'attribut suivant en cas de succès, et sinon, une erreur.

[ldap_next_attribute\(\)](#) sert à lire tous les attributs d'une entrée. Le pointeur interne est géré par *ber_identifiant*.

Il est passé par référence à la fonction. Le premier appel à [ldap_next_attribute\(\)](#) est fait avec le *result_entry_identifiant* retourné par [ldap_first_attribute\(\)](#).

Voir aussi [ldap_get_attributes\(\)](#).

[10.35.25 ldap_next_entry](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_next_entry](#) (int *link_identifiant*, int *result_entry_identifiant*)

Retourne l'identifiant de l'entrée suivante, dans le résultat qui a été initialisé par [ldap_first_entry\(\)](#). Si il n'y a plus d'entrée, retourne FALSE.

[ldap_next_entry\(\)](#) sert à retrouver toutes les entrées qui sont stockées dans un résultat. Les appels successifs à [ldap_next_entry\(\)](#) retourneront les entrées une à une. Le premier appel à [ldap_next_entry\(\)](#) est fait après un appel à [ldap_first_entry\(\)](#) avec *result_entry_identifiant* retourné par [ldap_first_entry\(\)](#).

Voir aussi [ldap_get_entries\(\)](#).

[10.35.26 ldap_read](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_read](#) (int *link_identifiant*, string *base_dn*, string *filter*, array *attributes*)

Retourne un identifiant de résultat en cas de succès, et FALSE sinon.

[ldap_read\(\)](#) effectue une recherche avec le filter *filter* dans le dossier *base_dn* et avec l'option LDAP_SCOPE_BASE (recherche limitée au dossier, ou récursive). Cela revient à lire une entrée dans un dossier.

Les filtres vides ne sont pas autorisés. Si vous souhaitez lire toutes les informations d'un dossier, utiliser le filtre suivant : "objectClass=*". Si vous savez quel est le type des entrées dans le dossier que vous fouillez, vous pouvez aussi adapter ce filter de la façon suivante "objectClass=inetOrgPerson".

Cette fonction dispose d'un quatrième argument optionnel. Reportez vous à [ldap_search\(\)](#).

[10.35.27 ldap_search](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_search](#) (int *link_identifiant*, string *base_dn*, string *filter*, array *attributes*)

Retourne un identifiant de résultat en cas de succès, et FALSE sinon.

[ldap_search\(\)](#) effectue une recherche avec le filtre *filter* dans le dossier *base_dn*, et avec l'option de récursivité LDAP_SCOPE_SUBTREE. Cela revient à rechercher dans toute la base sous le dossier *base_dn*. Le quatrième paramètre est optionnel, et peut être ajouté pour restreindre les attributs et les valeurs retournées. Il est beaucoup plus efficace que la méthode qui consiste à lire tous les attributs et leur valeurs associées. L'utilisation de ce quatrième paramètre est encouragée.

Le quatrième paramètre est un tableau de chaînes, qui contient les attributs désirés, array("mail","sn","cn")
 Notez que le nd *base_dn* est toujours retourné, quelques que soient les attributs demandés.

Notez que certains serveurs sont configurés pour limiter le nombre de résultats. Si cela arrive, le serveur indiquera qu'il n'a transféré qu'une partie du résultat.

La chaîne de filtre peut être simple ou complexe. Elle utilise les opérateurs booléens au même format que celui décrit dans les documentations LDAP. (Allez voir celle de [Netscape Directory SDK](#) pour plus d'informations sur les filtres).

L'exemple suivant récupère toutes les unités organisationnelles, le nom, prénom et email, dans la société "Ma Société" où le nom et prénom contiennent la sous-chaîne \$person. Cet exemple utilise un filtre booléen pour indiquer au serveur qu'il doit rechercher des informations dans plusieurs attributs. **Recherche LDAP**

```
// $ds est un identifiant valide de connexion à un serveur LDAP
// $person est tout ou une partir d'un nom
$dn = "o=Ma Société, c=Fr";
$filter="(|(sn=$person*)(givenname=$person*))";
$justthese = array( "ou", "sn", "givenname", "mail");
$sr=ldap_search($ds, $dn, $filter, $justthese);
$info = ldap_get_entries($ds, $sr);
print $info["count"]." Entrées retournées.<p>;
```

[10.35.28 ldap_unbind](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_unbind](#) (int *link_identifieur*)

Retourne TRUE en cas de succès, et FALSE sinon.

[ldap_unbind\(\)](#) termine la liaison avec le serveur LDAP.

[10.35.29 ldap_err2str](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ldap_err2str](#) (int *errno*)

Retourne un message d'erreur.

Cette fonction retourne le message d'erreur lié au numér d'erreur *errno*. Même si les numéros d'erreur LDAP sont standardisés, différentes librairies retournent différents messages, ou parfois, des messages en langue locale. Ne vous fiez pas au message d'erreur, mais bien au numéro d'erreur.

Voir aussi [ldap_errno\(\)](#) et [ldap_error\(\)](#). **Enumérer tous les messages d'erreur LDAP**

<?php

```

for($i=0; $i<100; $i++) {
printf("Error $i: %s<br>\n", ldap_str2err($i));
}
?>

```

[10.35.30 ldap_errno](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_errno](#)(int *link_id*)

Retourne le numéro d'erreur LDAP généré par la dernière commande.

Cette fonction retourne le numéro d'erreur standard, généré par la dernière commande LDAP, pour la connexion *link_id*. Ce numéro peut être converti en message textuel avec [ldap_err2str\(\)](#).

A moins que vous n'abaissiez suffisamment le niveau d'erreur dans ``php.ini'` (ou ``php3.ini'`), ou que vous ne préfixiez vos commandes LDAP avec (at) pour supprimer les affichages, les erreurs LDAP s'afficheront aussi dans le code PHP. *Générer et intercepter une erreur*

```

<?php
// Cet exemple contient une erreur, que nous allons intercepter.
$ld = ldap_connect("localhost");
$bind = ldap_bind($ld);
// Erreur de syntaxe dans l'expression du filtre (errno 87),
// ce doit être "objectclass=*"
$res = @ldap_search($ld, "o=Myorg, c=DE", "objectclass");
if (!$res) {
printf("LDAP-Errno: %s<br>\n", ldap_erro($ld));
printf("LDAP-Error: %s<br>\n", ldap_error($ld));
die("Argh!<br>\n");
}
$info = ldap_get_entries($ld, $res);
printf("%d entrées trouvées.<br>\n", $info["count"]);
?>

```

Voir aussi [ldap_err2str\(\)](#) et [ldap_error\(\)](#).

[10.35.31 ldap_error](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ldap_error](#)(int *link_id*)

Retourne un message d'erreur.

Cette fonction retourne le message d'erreur lié à la connexion *link_id*. Même si les numéros d'erreur LDAP sont standardisés, différentes bibliothèques retournent différents messages, ou parfois, des messages en langue locale. Ne vous fiez pas au message d'erreur, mais bien au numéro d'erreur.

A moins que vous n'abaissiez suffisamment le niveau d'erreur dans ``php.ini'` (ou ``php3.ini'`), ou que vous ne préfixiez vos commandes LDAP avec (at) pour supprimer les affichages, les erreurs LDAP s'afficheront aussi dans le code PHP.

Voir aussi [ldap_err2str\(\)](#) et [ldap_errno\(\)](#).

10.36 Fonction mail

[\[Notes en ligne\]](#)

`mail()` envoie du courrier électronique.

10.36.1 mail

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool `mail`(string *to*, string *subject*, string *message*, string *additional_headers*)

`mail()` poste automatiquement le message *message* à destination de *to*. Les destinataires multiples doivent être séparés par des virgules.

Envoi de mail.

```
mail("rasmus@lerdorf.on.ca", "Mon Sujet", "Ligne 1\nLigne 2\nLigne 3");
```

Le quatrième argument passé sera inséré à la fin de l'entête. Typiquement, cela permet d'insérer des entêtes supplémentaires. Les entêtes multiples doivent être séparés par des virgules.

Envoi de eMail avec des entêtes supplémentaires.

```
mail("nobody@aol.com", "Le sujet", $message,
     "From: webmaster@$SERVER_NAME\nReply-To: webmaster@$SERVER_NAME\nX-Mailer: PHP/" . phpversion());
```

Vous pouvez aussi utiliser des techniques simples de concaténations de chaînes pour construire des messages complexes : *Envoi de mail complexe.*

```
/* destinataire */
$recipient .= "Mary <mary@u.college.edu>" . ", " ; //remarquez les virgules
$recipient .= "Kelly <kelly@u.college.edu>" . ", " ;
$recipient .= "ronabop@php.net";
/* sujet */
$subject = "Rappel des anniversaires du mois d'aout";
/* message */
$message .= "Le mail suivant inclus une table au format ASCII\n";
$message .= "Jour \t\tMois \t\tAn\n";
$message .= "3 \t\tAou \t\t1970\n";
$message .= "17\t\tAou \t\t1973\n";
/* Vous pouvez ajouter une signature */
$message .= "--\r\n"; //Délimiteur de signature
$message .= "Rappel d'anniversaire : copyleft par public domain";
/* d'autres entêtes : errors, From cc's, bcc's, etc */
$headers .= "From: Rappel d'anniversaire <birthday@php.net>\n";
$headers .= "X-Sender: <birthday@php.net>\n";
$headers .= "X-Mailer: PHP\n"; // mailleur
$headers .= "X-Priority: 1\n"; // Message urgent!
$headers .= "Return-Path: <birthday@php.net>\n"; // Rechemin de retour pour les erreurs
$headers .= "Content-Type: text/html; charset=iso-8859-1\n" // Type MIME
```

```
$headers .= "cc:birthdayarchive@php.net\n"; // Champs CC
$headers .= "bcc:birthdaycheck@php.net, birthdaygifts@php.net\n"; // Champs BCCs
/* et hop, à la poste */
mail($recipient, $subject, $message, $headers);
```

10.36.2 ezmlm_hash

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ezmlm_hash](#)(string *addr*)

[ezmlm_hash\(\)](#) calcule la valeur de hash, nécessaire lors de la gestion de liste de diffusions EZMLM dans une base de données MySQL.

Calcul du hash et enregistrement d'un utilisateur

```
$user = "kris@koehtopp.de";
$hash = ezmlm_hash ($user);
$query = sprintf ("INSERT INTO sample VALUES (%s, '%s')", $hash, $user);
$db->query($query); // utilisation de l'interface PHPLIB
```

10.37 Fonctions mathématiques

[\[Notes en ligne\]](#)

10.37.1 Introduction

[\[Notes en ligne\]](#)

Ces fonctions ne sont capables de manipuler que des entiers double, ou des long. Si vous avez besoin de manipuler des nombres plus grands, reportez vous à [10.4 Fonctions mathématiques sur des nombres de taille arbitraire](#).

10.37.1.1 Constantes mathématiques

[\[Notes en ligne\]](#)

Les valeurs suivantes sont définies comme des constantes dans PHP:

| Constante | Valeur | Description |
|-----------|------------------------|-----------------|
| M_PI | 3.14159265358979323846 | La valeur de PI |

10.37.2 Abs

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [abs](#) (mixed *number*)

Retourne la valeur absolue du nombre *number*. Si le nombre est de type float, le type retourné est float, sinon, c'est integer (entier).

[10.37.3 Acos](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [acos](#) (float *arg*)

Retourne l'arc cosinus de *arg* (arg en radians).
Voir aussi [asin\(\)](#) et [atan\(\)](#).

[10.37.4 Asin](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [asin](#) (float *arg*)

Retourne l'arc sinus de *arg* (arg en radians).
Voir aussi [acos\(\)](#) et [atan\(\)](#).

[10.37.5 Atan](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [atan](#) (float *arg*)

Retourne l'arc tangent de *arg* (arg en radians).
Voir aussi [acos\(\)](#) et [atan\(\)](#).

[10.37.6 Atan2](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [atan2](#) (float *y*, float *x*)

Retourne l'arc tangent de deux variables *x* et *y*. La formule est : " arc tangent (*y / x*) ", et les signes des arguments sont utilisés pour déterminer le quadrant du résultat.
Cette fonction retourne un résultat en radians, entre $-\text{PI}$ et PI (inclus).
Voir aussi [acos\(\)](#) et [atan\(\)](#).

[10.37.7 base_convert](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [base_convert](#) (string *number*, int *frombase*, int *tobase*)

Retourne une chaîne contenant l'argument *number* représenté dans la base *tobase*. La base de représentation de *number* est donnée par *frombase*. *frombase* et *tobase* doivent être compris entre 2 et 36, inclus. Les chiffres supérieurs à 10 des bases supérieures à 10 seront représentées par les lettres de a à z, avec a = 10 et z

= 36. *base_convert()*

```
$binary = base_convert($hexadecimal, 16, 2);
```

10.37.8 BinDec

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [bindec\(\)](#)(string *binary_string*)

Retourne la conversion d'un nombre binaire représenté par la chaîne *binary_string* en décimal. [bindec\(\)](#) convertit un nombre binaire en décimal. Le plus grand nombre convertible a 31 bits à 1, soit 2147483647 en décimal. Voir aussi [decbin\(\)](#).

10.37.9 Ceil

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ceil\(\)](#)(float *number*)

Retourne l'entier supérieur du nombre *number*. Utiliser [ceil\(\)](#) sur un entier ne sert à rien. NOTE: [ceil\(\)](#) sous PHP/FI 2 renvoyait un float. Utilisez: \$new = (double)ceil(\$number); pour retrouver le comportement traditionnel. Voir aussi [floor\(\)](#) et [round\(\)](#).

10.37.10 Cos

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [cos\(\)](#)(float *arg*)

Retourne le cosinus de arg (arg en radians). Voir aussi [sin\(\)](#) et [tan\(\)](#).

10.37.11 DecBin

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [decbin\(\)](#)(int *number*)

Retourne une chaîne contenant la représentation binaire de l'entier donné en argument. Le plus grand nombre pouvant être converti est 2147483647 en décimal, ce qui donne une série de 31 uns (1). Voir aussi [bindec\(\)](#).

[10.37.12 DecHex](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dechex](#) (int *number*)

Retourne une chaîne contenant la représentation hexadécimale du nombre *number*. Le nombre le plus grand qui puisse être converti est 2147483647 en décimal, ce qui donnera "7fffffff".

Voir aussi [hexdec\(\)](#).

[10.37.13 DecOct](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [decoct](#) (int *number*)

Retourne une chaîne contenant la représentation octale du nombre donné *number*. Le nombre le plus grand qui puisse être converti est 2147483647 en décimal, ce qui donnera "1777777777".

Voir aussi [octdec\(\)](#).

[10.37.14 Exp](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [exp](#) (float *arg*)

Retourne l'exponentielle de *arg*, c'est à dire e élevé à la puissance *arg*.

Voir aussi [pow\(\)](#).

[10.37.15 Floor](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [floor](#) (float *number*)

Retourne l'entier inférieur du nombre *number*. Utiliser [floor\(\)](#) sur un entier ne sert à rien.

NOTE: [floor\(\)](#) sous PHP/FI renvoyait un float. Utilisez: \$new = (double)floor(\$number); pour retrouver le comportement traditionnel.

Voir aussi [ceil\(\)](#) et [round\(\)](#).

[10.37.16 getrandmax](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getrandmax](#) (void)

Retourne la plus grande valeur aléatoire possible retournée par [rand\(\)](#).

Voir aussi [rand\(\)](#), [srand\(\)](#), [mt_rand\(\)](#), [mt_srand\(\)](#) et [mt_getrandmax\(\)](#).

10.37.17 HexDec

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hexdec](#)(string *hex_string*)

Retourne une chaîne contenant la représentation décimal du nombre *hex_string*. Le nombre le plus grand qui puisse être converti est 7fffffff en décimal, ce qui donne "2147483647".

Voir aussi the [dechex\(\)](#).

10.37.18 Log

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [log](#)(float *arg*)

Retourne le logarithme naturel de *arg*.

10.37.19 Log10

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [log10](#)(float *arg*)

Retourne le logarithme en base 10 de *arg*.

10.37.20 max

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [max](#)(mixed *arg1*, mixed *arg2*, mixed *argn*)

[max\(\)](#) retourne la plus grande valeur numérique parmi les valeurs passées en paramètre.

Si le premier paramètre est un tableau, [max\(\)](#) retourne la plus grande valeur de ce tableau. Si le premier paramètre est un entier, une chaîne ou un double, [max\(\)](#) requiert au moins deux paramètres, et retournera alors le plus grand d'entre eux. Le nombre d'argument est alors illimité.

Si au moins une valeur est de type double, elle seront toutes traitées comme des doubles, et un double sera retourné. Si aucune valeur n'est de type double, elles seront traitées comme des entiers, et un entier sera retourné.

10.37.21 min

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [min](#)(mixed *arg1*, mixed *arg2*, mixed *argn*)

[min\(\)](#) retourne la plus petite valeur numérique parmi les valeurs passées en paramètre.

Si le premier paramètre est un tableau, [min\(\)](#) retourne la plus petite valeur de ce tableau. Si le premier paramètre est un entier, une chaîne ou un double, [min\(\)](#) requiert au moins deux paramètres, et retournera alors le plus petit d'entre eux. Le nombre d'argument est alors illimité.

Si au moins une valeur est de type double, elle seront toutes traitées comme des doubles, et un double sera retourné. Si aucune valeur n'est de type double, elles seront traitées comme des entiers, et un entier sera

retourné.

[10.37.22 mt_rand](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mt_rand](#)(*int min* , *int max*)

De nombreux générateurs de nombre aléatoires provenant de vieilles bibliothèques libc ont des comportements douteux et sont très lents. Par défaut, PHP utilise le générateur de nombres aléatoires de libc avec la fonction [rand\(\)](#). [mt_rand\(\)](#) est une fonction de remplacement, pour cette dernière. Elle utilise un générateur de nombre aléatoire de caractéristique connue, le " Mersenne Twister ", qui va produire des nombres utilisables en cryptographie, et qui est 4 fois plus rapide que la fonction standard libc. La "Homepage of the Mersenne Twister " est <http://www.math.keio.ac.jp/~matumoto/emt.html>, une version optimisée des sources de MT est disponible à <http://www.scp.syr.edu/~marc/hawk/twister.html>. Appelé sans les arguments optionnels *min*, *max*, [mt_rand\(\)](#) retourne un nombre pseudo-aléatoire, entre 0 et RAND_MAX. Pour obtenir un nombre entre 5 et 15 (inclus), il faut utiliser `mt_rand(5,15)`.

N'oubliez pas d'initialiser le générateur de nombre aléatoire avec [mt_srand\(\)](#).

Note : *Dans les versions antérieure à la 3.0.7 la signification du paramètre **max** était "longueur". Pour avoir le même résultat, il faut utiliser `mt_rand (5, 11)` pour obtenir un nombre aléatoire entre 5 et 15.*

Voir aussi [mt_srand\(\)](#), [mt_getrandmax\(\)](#), [srand\(\)](#), [rand\(\)](#) et [getrandmax\(\)](#).

[10.37.23 mt_srand](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [mt_srand](#)(int *seed*)

Initialise une meilleure valeur aléatoire avec *seed*.

```
// initialise avec les microsecondes depuis la dernière seconde entière
mt_srand((double)microtime()*1000000);
$randval = mt_rand();
```

Voir aussi [mt_rand\(\)](#), [mt_getrandmax\(\)](#), [srand\(\)](#), [rand\(\)](#) et [getrandmax\(\)](#).

[10.37.24 mt_getrandmax](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mt_getrandmax](#)(void)

Retourne la plus grand valeur aléatoire possible que peut retourner [mt_rand\(\)](#).

Voir aussi [mt_rand\(\)](#), [mt_srand\(\)](#), [rand\(\)](#), [srand\(\)](#) et [getrandmax\(\)](#).

[10.37.25 number_format](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [number_format](#)(float *number*, int *decimals*, string *dec_point*, string *thousands_sep*)

[number_format\(\)](#) retourne une chaîne représentant *number* formaté. Cette fonction accepte un, deux ou 4 paramètres (mais pas trois).

Si un seul paramètre est donné, *number* sera formaté sans décimale, mais avec une virgule entre chaque série de 1000.

Avec deux paramètres, *number* sera formaté avec *decimals* décimales et un point ("."), une virgule entre chaque série de 1000.

Avec quatre paramètres, *number* sera formaté avec *decimals* décimales et *dec_point* à la place du point, une virgule entre chaque série de 1000.

[10.37.26 OctDec](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [octdec](#)(string *octal_string*)

Retourne une chaîne contenant la représentation décimale du nombre *octal_string*. Le nombre le plus grand qui puisse être converti est 1777777777 en décimal, ce qui donnera "2147483647".

Voir aussi [decoct\(\)](#).

[10.37.27 pi](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

double [pi](#)(void)

Retourne la valeur de pi.

[10.37.28 pow](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [pow](#)(float *base*, float *exp*)

Retourne *base* élevé à la puissance *exp*.

Voir aussi [exp\(\)](#).

[10.37.29 rand](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [rand](#)(*int min* , *int max*)

Appelée sans les options *min* et *max*, [rand\(\)](#) retourne un nombre pseudo-aléatoire entre 0 et RAND_MAX. Si vous voulez un nombre aléatoire entre 5 et 15 (inclus), par exemple, utilisez [rand\(5, 15\)](#).

N'oubliez pas d'initialiser le générateur de nombres aléatoires avec [srand\(\)](#).

Note : Dans les versions antérieure à la 3.0.7 la signification du paramètre *max* était longueur. Pour avoir le même résultat, il faut utiliser [mt_rand\(5, 11\)](#) pour obtenir un nombre aléatoire entre 5 et 15.

Voir aussi [srand\(\)](#), [getrandmax\(\)](#), [mt_rand\(\)](#), [mt_srand\(\)](#) et [mt_getrandmax\(\)](#).

[10.37.30 round](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

double [round](#) (double *val*)

Retourne la valeur arrondie de *val*.

```
$foo = round( 3.4 ); // $foo == 3.0
$foo = round( 3.5 ); // $foo == 4.0
$foo = round( 3.6 ); // $foo == 4.0
```

Voir aussi [ceil\(\)](#) et [floor\(\)](#).

[10.37.31 Sin](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [sin](#) (float *arg*)

Retourne le sinus de *arg* (arg in radians).

Voir aussi [cos\(\)](#) et [tan\(\)](#).

[10.37.32 Sqrt](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [sqrt](#) (float *arg*)

Retourne la racine carrée de *arg*.

[10.37.33 srand](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [srand](#) (int *seed*)

Initialise le générateur de nombres aléatoires avec *seed*.

```
// initialise avec les microsecondes depuis la dernière seconde entière
sret((double)microtime()*1000000);
$randval = rand();
```

Voir aussi [rand\(\)](#), [getrandmax\(\)](#), [mt_rand\(\)](#), [mt_srand\(\)](#) et [mt_getrandmax\(\)](#).

[10.37.34 Tan](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [tan](#) (float *arg*)

Retourne la tangente de *arg* (arg en radians).
Voir aussi [sin\(\)](#) et [cos\(\)](#).

10.38 Fonctions MCAL

[\[Notes en ligne\]](#)

MCAL signifie Modular Calendar Access Library (bibliothèque calendaire modulaire).

Libmcal est une bibliothèque C de calendriers. Elle est écrite pour être très modulaire, et dispose de nombreux modules. MCAL est l'équivalent de IMAP pour les calendriers.

Avec mcal, un calendrier peut être ouvert comme une boîte aux lettres. Les calendriers peuvent être des fichiers locaux, ou bien être sur des serveurs ICAP distants, ou encore tout autre format supporté par la bibliothèque.

Les événements peuvent être lus, sélectionnés et enregistrés. Il y a aussi la possibilité d'ajouter des alarmes, et de placer des événements récurrents.

Avec libmcal, les serveurs centralisés peuvent être accédés et utilisés, et remplacent avantageusement tout développement spécifique de base de données.

Pour faire fonctionner cette bibliothèque, vous devez compiler PHP avec l'option `--with-mcal`. Il vous faudra alors avoir installé la bibliothèque mcal. Téléchargez la dernière version à <http://mcal.chek.com/> et compilez la, puis installez la.

Les constantes suivantes sont définies lorsque mcal est utilisée : MCAL_SUNDAY, MCAL_MONDAY, MCAL_TUESDAY, MCAL_WEDNESDAY, MCAL_THURSDAY, MCAL_FRIDAY, MCAL_SATURDAY, MCAL_RECUR_NONE, MCAL_RECUR_DAILY, MCAL_RECUR_WEEKLY, MCAL_RECUR_MONTHLY_MDAY, MCAL_RECUR_MONTHLY_WDAY, MCAL_RECUR_YEARLY, MCAL_JANUARY, MCAL_FEBRUARY, MCAL_MARCH, MCAL_APRIL, MCAL_MAY, MCAL_JUNE, MCAL_JULY, MCAL_AUGUST, MCAL_SEPTEMBER, MCAL_OCTOBER, MCAL_NOVEMBER, et MCAL_DECEMBER. La plupart des fonctions utilisent une structure d'événement interne, qui est unique pour chaque connexion. Cela évite d'avoir à passer des objets de grande taille entre les fonctions. Il y a des accesseurs bien pratiques pour créer, initialiser et lire des objets événements.

10.38.1 mcal_open

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_open](#)(string *calendar*, string *username*, string *password*, string *options*)

Retourne un flot MCAL en cas de succès, et FALSE en cas d'erreur.

[mcal_open\(\)](#) ouvre une connexion MCAL au serveur *calendar*. Si *options* est spécifié, passe aussi *options* à la boîte aux lettres (???). La structure interne du flot MCAL est initialisée à la connexion.

10.38.2 mcal_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_close](#)(int *mcal_stream*, int *flags*)

Ferme la connexion.

10.38.3 mcal_fetch_event

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [mcal_fetch_event](#) (int *mcal_stream*, int *event_id*, int *options*)

[mcal_fetch_event\(\)](#) recherche un événement dans le calendrier spécifié par *id*.

Retourne un objet événement dont les attributs sont :

- int id – ID de l'événement.
- int public – TRUE si l'événement est public, FALSE si il est privé.
- string category – Catégorie de l'événement.
- string title – Titre de l'événement.
- string description – Description de l'événement.
- int alarm – Nombre de minutes avant d'envoyer une alerte pour cet événement.
- object start – Objet contenant une date et une heure.
- object end – Objet contenant une date et une heure.
- int recur_type – type de récurrence
- int recur_interval – intervalle de récurrence
- datetime recur_enddate – date de fin de récurrence
- int recur_data – données de récurrence

Tous les objets de date et heure sont construits comme suit :

- int year – année
- int month – mois
- int mday – jour du mois
- int hour – heure
- int min – minutes
- int sec – secondes
- int alarm – nombre de minutes avant de déclencher l'alarme

10.38.4 mcal_list_events

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mcal_list_events](#) (int *mcal_stream*, int *begin_year* , int *begin_month* , int *begin_day* , int *end_year* , int *end_month* , int *end_day*)

Retourne un tableau d'identifiants d'événements, compris entre deux dates.

[mcal_list_events\(\)](#) prend une date de début et une date de fin. Un tableau d'identifiants est retourné.

10.38.5 mcal_append_event

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_append_event](#) (int *mcal_stream*)

[mcal_append_event\(\)](#) enregistre l'événement global dans le calendrier MCAL *mcal_stream*.

Retourne l'uid de l'enregistrement ainsi inséré.

10.38.6 mcal_store_event

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_store_event](#)(int *mcal_stream*)

[mcal_store_event\(\)](#) enregistre l'événement global dans le calendrier MCAL *mcal_stream*.
Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

10.38.7 mcal_delete_event

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_delete_event](#)(int *uid*)

[mcal_delete_event\(\)](#) efface l'événement d'identifiant *uid*.
Retourne TRUE.

10.38.8 mcal_snooze

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_snooze](#)(int *uid*)

[mcal_snooze\(\)](#) éteint l'alarme de l'événement identifié par l'UID *uid*.
Retourne TRUE.

10.38.9 mcal_list_alarms

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mcal_list_events](#)(int *mcal_stream*, int *begin_year* , int *begin_month* , int *begin_day* , int *end_year* , int *end_month* , int *end_day*)

Retourne un tableau d'identifiants, qui ont une alarme de prévue à la date *alarm_date*. Si seul le flot MCAL est donné, la date de début et de fin de la structure globale sera utilisée.

[mcal_list_events\(\)](#) prend une date, et retourne un tableau d'identifiants.

10.38.10 mcal_event_init

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_init](#)(int *stream*)

[mcal_event_init\(\)](#) initialise la structure globale d'un flot. Cela remet tous les éléments de la structure à 0, ou à leur valeur par défaut.

Retourne TRUE.

[10.38.11 mcal_event_set_category](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_category](#)(int *stream*, string *category*)

[mcal_event_set_category\(\)](#) fixe la catégorie de la structure globale à la valeur de *category*.
Retourne TRUE.

[10.38.12 mcal_event_set_title](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_title](#)(int *stream*, string *title*)

[mcal_event_set_title\(\)](#) fixe le titre de la structure globale à la valeur de *title*.
Retourne TRUE.

[10.38.13 mcal_event_set_description](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_description](#)(int *stream*, string *description*)

[mcal_event_set_description\(\)](#) fixe la catégorie de la structure globale à la valeur de *description*.
Returns TRUE.

[10.38.14 mcal_event_set_start](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_start](#)(int *stream*, int *year*, int *month*, int *day* , int *hour* , int *min* , int *sec*)

[mcal_event_set_start\(\)](#) fixe la date de début de la structure globale.
Retourne TRUE.

[10.38.15 mcal_event_set_end](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_end](#)(int *stream*, int *year*, int *month*, int *day* , int *hour* , int *min* , int *sec*)

[mcal_event_set_end\(\)](#) fixe la date de fin de la structure globale.
Returns TRUE.

[10.38.16 mcal_event_set_alarm](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_alarm](#)(int *stream*, int *alarm*)

[mcal_event_set_alarm\(\)](#) fixe l'alarme de la structure globale, à un nombre de minutes avant déclenchement. Retourne TRUE.

[10.38.17 mcal_event_set_class](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_class](#)(int *stream*, int *class*)

[mcal_event_set_class\(\)](#) Fixe la classe de la structure globale. La classe vaut 0 pour public, et 1 pour privée. Retourne TRUE.

[10.38.18 mcal_is_leap_year](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_is_leap_year](#)(int *year*)

[mcal_is_leap_year\(\)](#) retourne 1 si l'année *year* est bissextile, et 0 sinon.

[10.38.19 mcal_days_in_month](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_days_in_month](#)(int *month*, int *leap year*)

[mcal_days_in_month\(\)](#) retourne le nombre de jour du mois *month*, et prend en compte le fait que l'année est bissextile avec le paramètre *leap year*.

[10.38.20 mcal_date_valid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_date_valid](#)(int *year*, int *month*, int *day*)

[mcal_date_valid\(\)](#) retourne TRUE si la date (constituée par l'année *year*, le mois *month* et la date *day*) est valide, et FALSE sinon.

[10.38.21 mcal_time_valid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_time_valid](#)(int *hour*, int *minutes*, int *seconds*)

[mcal_time_valid\(\)](#) retourne TRUE si l'heure (constituée par l'heure *hour*, les minutes *minutes* et les secondes *seconds*) est une heure valide, et FALSE sinon.

[10.38.22 mcal_day_of_week](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_day_of_week](#)(int *year*, int *month*, int *day*)

[mcal_day_of_week\(\)](#) retourne le jour de la semaine, pour la date constituée par l'année *year*, le mois *month* et la date *day*.

[10.38.23 mcal_day_of_year](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_day_of_year](#)(int *year*, int *month*, int *day*)

[mcal_day_of_year\(\)](#) retourne le numéro de jour dans l'année pour la date constituée par l'année *year*, le mois *month* et la date *day*.

[10.38.24 mcal_date_compare](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_date_compare](#)(int *a_year*, int *a_month*, int *a_day*, int *b_year*, int *b_month*, int *b_day*)

[mcal_date_compare\(\)](#) compare les deux dates données, et retourne <0, 0, >0 si a<b, a==b, a>b respectivement.

[10.38.25 mcal_next_recurrence](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_next_recurrence](#)(int *stream*, int *weekstart*, array *next*)

[mcal_next_recurrence\(\)](#) retourne un objet contenant la prochaine date de l'événement, ou la date de l'événement suivant la date. Retourne un objet date vide si l'événement n'a pas de récurrence, ou si quelque chose est invalide. Utilisez *weekstart* pour déterminer le premier jour.

[10.38.26 mcal_event_set_recur_none](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_recur_none](#)(int *stream*)

[mcal_event_set_recur_none\(\)](#) supprime la récurrence de la structure globale (event->recur_type est mis à MCAL_RECUR_NONE).

[10.38.27 mcal_event_set_recur_daily](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_recur_daily](#)(int *stream*, int *year*, int *month*, int *day*, int *interval*)

[mcal_event_set_recur_daily\(\)](#) fixe la récurrence quotidienne de la structure globale, jusqu'à la date passée en paramètre. (la date de début est celle de la structure).

[10.38.28 mcal_event_set_recur_weekly](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_recur_weekly](#) (int *stream*, int *year*, int *month*, int *day*, int *interval*, int *weekdays*)

[mcal_event_set_recur_weekly\(\)](#) fixe la récurrence hebdomadaire de la structure globale, jusqu'à la date passée en paramètre. (la date de début est celle de la structure).

[10.38.29 mcal_event_set_recur_monthly_mday](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_recur_monthly_mday](#) (int *stream*, int *year*, int *month*, int *day*, int *interval*)

[mcal_event_set_recur_monthly_mday\(\)](#) fixe la récurrence de la structure globale, jusqu'à la date passée en paramètre. (la date de début est celle de la structure).

[10.38.30 mcal_event_set_recur_monthly_wday](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_recur_monthly_wday](#) (int *stream*, int *year*, int *month*, int *day*, int *interval*)

[mcal_event_set_recur_monthly_wday\(\)](#) fixe la récurrence mensuelle de la structure globale, jusqu'à la date passée en paramètre. (la date de début est celle de la structure).

[10.38.31 mcal_event_set_recur_yearly](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_recur_yearly](#) (int *stream*, int *year*, int *month*, int *day*, int *interval*)

[mcal_event_set_recur_yearly\(\)](#) fixe la récurrence annuelle de la structure globale, jusqu'à la date passée en paramètre. (la date de début est celle de la structure).

[10.38.32 mcal_fetch_current_stream_event](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_fetch_current_stream_event](#) (int *stream*)

[mcal_fetch_current_stream_event\(\)](#) retourne la structure de la date du flot courant sous la forme d'un objet, qui contient :

- int id – ID de l'événement.
- int public – TRUE si l'événement est public, FALSE si il est privé.
- string category – Catégorie de l'événement.
- string title – Titre de l'événement.
- string description – Description de l'événement.
- int alarm – Nombre de minutes avant d'envoyer une alerte pour cet événement.
- object start – Objet contenant une date et une heure.
- object end – Objet contenant une date et une heure.
- int recur_type – type de récurrence
- int recur_interval – intervalle de récurrence
- datetime recur_enddate – date de fin de récurrence
- int recur_data – données de récurrence

Tous les objets de date et heure sont construits comme suit :

- int year – année
- int month – mois
- int mday – jour du mois
- int hour – heure
- int min – minutes
- int sec – secondes
- int alarm – nombre de minutes avant de déclencher l'alarme

10.39 Cryptage

[\[Notes en ligne\]](#)

Ces fonctions utilisent [mcrypt](#).

Ces fonctions permettent d'accéder à la librairie mcrypt, qui dispose d'une grande variété d'algorithmes de cryptage, tels que DES, TripleDES, Blowfish (par défaut), 3-WAY, SAFER-SK64, SAFER-SK128, TWOFISH, TEA, RC2 et GOST en modes CBC, OFB, CFB et ECB. De plus, elle accepte aussi RC6 et IDEA qui sont considérés comme "non libre".

Si vous compilez PHP avec la librairie libmccrypt 2.4.x, les algorithmes suivants sont supportés : CAST, LOKI97, RIJNDAEL, SAFERPLUS, SERPENT ainsi que les chiffrements suivants : ENIGMA (cryptage), PANAMA, RC4 et WAKE. Avec libmccrypt 2.4.x un autre mode de chiffrement est disponible : nOFB.

Pour l'utiliser, téléchargez la librairie libmccrypt-x.x.tar.gz grâce par [ici](#) et suivez les instructions d'installations incluses. Vous aurez aussi besoin de compiler PHP avec le paramètre `--with-mcrypt` pour activer cette extension.

Mcrypt permet de crypter et de décrypter, en utilisant les méthodes mentionnées ci-dessus. Les 4 commandes importantes [mccrypt_cfb\(\)](#), [mccrypt_cbc\(\)](#), [mccrypt_ecb\(\)](#) et [mccrypt_ofb\(\)](#) peuvent toutes opérer en mode MCRYPT_ENCRYPT et MCRYPT_DECRYPT. ***Crypte une valeur avec un TripleDES, en mode ECB.***

```
<?php
$key = "Cette cle est ultra secreete";
$input = "Rencontrons nous dans notre place secreete a 9 h 00.";
$encrypted_data = mcrypt_ecb(MCRYPT_TriplesDES, $key, $input, MCRYPT_ENCRYPT);
?>
```

Cet exemple va retourner les données cryptées dans la variable \$encrypted_data.

Si vous avez compilé PHP avec libmcrypt 2.4.x, ces fonctions sont toujours disponibles, mais il est vivement conseillé d'utiliser les nouvelles fonctions avancées. *Encryption d'une valeur avec TripleDES sous 2.4.x en mode ECB*

```
<?php
$key = "Ceci est une vraie cle secrete";
$input = "Rendez vous à 9 heures, dans notre planque.";
$td = mcrypt_module_open (MCRYPT_TripleDES, "", MCRYPT_MODE_ECB, "");
$iv = mcrypt_create_iv (mcrypt_enc_get_iv_size ($td), MCRYPT_RAND);
mcrypt_generic_init ($td, $key, $iv);
$encrypted_data = mcrypt_generic ($td, $input);
mcrypt_generic_end ($td);
?>
```

Cet exemple va retourner les données cryptées dans la variable \$encrypted_data.

Mcrypt peut opérer en 4 modes de cryptage (CBC, OFB, CFB, and ECB). Nous allons présenter la technique d'utilisation de ces modes. Pour plus de références et de détails, reportez vous au livre suivant : Applied Cryptography par Schneier (ISBN 0-471-11709-9).

- ECB (electronic codebook) ECB (electronic codebook) est prévu pour des données aléatoires, telles que des clés. Etant donné que les données sont peu nombreuses et aléatoires, les inconvénients de l'ECB ont ici un effet négatif favorable.
- CBC (cipher block chaining) est spécialement pratique avec les fichiers dont la sécurité ECB n'est pas suffisante.
- CFB (cipher feedback) est la meilleure méthode pour crypter des flots d'octets, quand les octets doivent être encryptés un par un.
- OFB (output feedback) est comparable à CFB, mais peut être utilisé lorsque des erreurs ne doivent pas être propagées.
- nOFB (output feedback, in nbit) est comparable à OFB, mais plus sûr, car il opère avec la taille de bloc de l'algorithme.
- STREAM est un mode supplémentaire, pour permettre l'utilisation d' algorithmes tels que WAKE ou RC4.

PHP ne supporte par encore le cryptage des flots d'octets. Pour l'instant, PHP n'accepte que le cryptage de chaîne.

Pour obtenir la liste complète des modes de chiffrement, reportez vous aux derniers #define, dans le fichier `mcrypt.h`. En règle générale, vous pouvez accéder à une méthode de chiffrement avec l'option MCRYPT_nomDuChiffrement.

Voici une liste non exhaustive des modes de chiffrements de l'extension mcrypt. Si un chiffrement n'est pas dans cette liste, mais disponible dans la librairie, vous pouvez supposer que cette documentation est hors d'âge.

- MCRYPT_3DES
- MCRYPT_ARCFOUR_IV (libmcrypt 2.4.x seulement)
- MCRYPT_ARCFOUR (libmcrypt 2.4.x seulement)
- MCRYPT_BLOWFISH
- MCRYPT_CAST_128
- MCRYPT_CAST_256
- MCRYPT_CRYPT

- MCRYPT_DES
- MCRYPT_DES_COMPAT (libmcrypt 2.2.x seulement)
- MCRYPT_ENIGMA (libmcrypt 2.4.x seulement, alias de MCRYPT_CRYPT)
- MCRYPT_GOST
- MCRYPT_IDEA (payant)
- MCRYPT_LOKI97 (libmcrypt 2.4.x seulement)
- MCRYPT_MARS (libmcrypt 2.4.x seulement, payant)
- MCRYPT_PANAMA (libmcrypt 2.4.x seulement)
- MCRYPT_RIJNDAEL_128 (libmcrypt 2.4.x seulement)
- MCRYPT_RIJNDAEL_192 (libmcrypt 2.4.x seulement)
- MCRYPT_RIJNDAEL_256 (libmcrypt 2.4.x seulement)
- MCRYPT_RC2
- MCRYPT_RC4 (libmcrypt 2.2.x seulement)
- MCRYPT_RC6 (libmcrypt 2.4.x seulement)
- MCRYPT_RC6_128 (libmcrypt 2.2.x seulement)
- MCRYPT_RC6_192 (libmcrypt 2.2.x seulement)
- MCRYPT_RC6_256 (libmcrypt 2.2.x seulement)
- MCRYPT_SAFER64
- MCRYPT_SAFER128
- MCRYPT_SAFERPLUS (libmcrypt 2.4.x seulement)
- MCRYPT_SERPENT (libmcrypt 2.4.x seulement)
- MCRYPT_SERPENT_128 (libmcrypt 2.2.x seulement)
- MCRYPT_SERPENT_192 (libmcrypt 2.2.x seulement)
- MCRYPT_SERPENT_256 (libmcrypt 2.2.x seulement)
- MCRYPT_SKIPJACK (libmcrypt 2.4.x seulement)
- MCRYPT_TEAN (libmcrypt 2.2.x seulement)
- MCRYPT_THREEWAY
- MCRYPT_TRIPLEDES (libmcrypt 2.4.x seulement)
- MCRYPT_TWOFISH (Pour les anciennes versions de mcrypt 2.x versions, ou mcrypt 2.4.x)
- MCRYPT_TWOFISH128 (TWOFISHxxx sont disponibles avec les nouvelles versions de 2.x, mais pas dans les versions 2.4.x)
- MCRYPT_TWOFISH192
- MCRYPT_TWOFISH256
- MCRYPT_WAKE (libmcrypt 2.4.x seulement)
- MCRYPT_XTEA (libmcrypt 2.4.x seulement)

Vous devez (mode CFB et OFB) ou pouvez (mode CBC) fournir un vecteur d'initialisation (IV) pour ces modes de chiffrement. IV doit être unique, et avoir la même valeur au chiffrement et au déchiffrement. Pour des données qui seront enregistrées après encryptage, vous pouvez prendre le résultat d'une fonction telle que MD5, appliquée sur le nom du fichier. Sinon, vous pouvez envoyer IV avec les données chiffrées, (reportez vous au chapitre 9.3 de Applied Cryptography by Schneier (ISBN 0-471-11709-9) pour plus de détails sur le sujet).

[10.39.1 mcrypt_get_cipher_name](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_get_cipher_name](#) (int *cipher*)

string [mcrypt_get_cipher_name](#) (string *cipher*)

[mcrypt_get_cipher_name\(\)](#) retourne le nom du chiffrement utilisé.

[mcrypt_get_cipher_name\(\)](#) prend le numéro de chiffrement comme paramètre, et retourne le nom du chiffrement, ou FALSE, si ce chiffrement n'existe pas.

Exemple avec mcrypt_get_cipher_name

```
<?php
$cipher = MCRYPT_TripleDES;
print mcrypt_get_cipher_name($cipher);
?>
```

L'exemple ci dessus va donner :

TripleDES

10.39.2 mcrypt_get_block_size

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_get_block_size](#)(int *cipher*)

[mcrypt_get_block_size\(\)](#) sert à lire la taille de bloc du chiffrement *cipher*.

[mcrypt_get_block_size\(\)](#) prend comme argument le chiffrement *cipher* et retourne une taille en octets.

Voir aussi : [mcrypt_get_key_size\(\)](#).

10.39.3 mcrypt_get_key_size

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_get_key_size](#)(int *cipher*)

[mcrypt_get_key_size\(\)](#) sert à lire la taille de clé du chiffrement *cipher*.

[mcrypt_get_block_size\(\)](#) prend comme argument le chiffrement *cipher* et retourne une taille en octets.

Voir aussi: [mcrypt_get_block_size\(\)](#).

10.39.4 mcrypt_create_iv

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_create_iv](#)(int *size*, int *source*)

[mcrypt_create_iv\(\)](#) sert à créer un IV (vecteur d'initialisation).

[mcrypt_create_iv\(\)](#) prend deux arguments, *size* détermine la taille de IV, *source* spécifie la source de IV.

La source peut être MCRYPT_RAND (générateur de nombres aléatoires système),

MCRYPT_DEV_RANDOM (lecture des données depuis le fichier /dev/random) et

MCRYPT_DEV_URANDOM (lecture des données depuis le fichier /dev/urandom). Si vous utilisez

MCRYPT_RAND, assurez vous de bien appeler [srand\(\)](#) pour initialiser le générateur de nombres aléatoires.

Exemple avec mcrypt_create_iv

```
<?php
$cipher = MCRYPT_TripleDES;
$block_size = mcrypt_get_block_size($cipher);
$iv = mcrypt_create_iv($block_size, MCRYPT_DEV_RANDOM);
?>
```

10.39.5 mcrypt_cbc

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_cbc](#)(int *cipher*, string *key*, string *data*, int *mode*, string *iv*)

string [mcrypt_cbc](#)(string *cipher*, string *key*, string *data*, int *mode*, string *iv*)

La première syntaxe utilise libmcrypt 2.2.x, et la seconde utilise libmcrypt 2.4.x.

[mcrypt_cbc\(\)](#) encrypte ou décrypte (suivant le *mode* sélectionné) les données *data* avec le chiffrement *cipher* et la clé *key* en mode CBC et retourne la chaîne résultant.

Cipher est une des constantes MCRYPT_ciphername

Key est la clé fournie à l'algorithme. Elle doit être tenue secrète.

Data sont les données à traiter.

Mode vaut MCRYPT_ENCRYPT ou MCRYPT_DECRYPT.

IV est le vecteur d'initialisation (optionnel).

Voir aussi: [mcrypt_cfb\(\)](#), [mcrypt_ecb\(\)](#), et [mcrypt_ofb\(\)](#).

10.39.6 mcrypt_cfb

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_cfb](#)(int *cipher*, string *key*, string *data*, int *mode*, string *iv*)

string [mcrypt_cfb](#)(string *cipher*, string *key*, string *data*, int *mode*, string *iv*)

La première syntaxe utilise libmcrypt 2.2.x, et la seconde utilise libmcrypt 2.4.x.

[mcrypt_cfb\(\)](#) encrypte ou décrypte (suivant le *mode* sélectionné) les données *data* avec le chiffrement *cipher* et la clé *key* en mode CFB et retourne la chaîne résultant.

Cipher est une des constantes MCRYPT_ciphername

Key est la clé fournie à l'algorithme. Elle doit être tenue secrète.

Data sont les données à traiter.

Mode vaut MCRYPT_ENCRYPT ou MCRYPT_DECRYPT.

IV est le vecteur d'initialisation (optionnel).

Voir aussi: [mcrypt_cbc\(\)](#), [mcrypt_ecb\(\)](#), et [mcrypt_ofb\(\)](#).

10.39.7 mcrypt_ecb

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_ecb](#)(int *cipher*, string *key*, string *data*, int *mode*)

string [mdecrypt_ecb](#) (string *cipher*, string *key*, string *data*, int *mode*, string *iv*)

La première syntaxe utilise libmcrypt 2.2.x, et la seconde utilise libmcrypt 2.4.x.

[mdecrypt_ecb\(\)](#) encrypte ou décrypte (suivant le *mode* sélectionné) les données *data* avec le chiffrement *cipher* et la clé *key* en mode CFB et retourne la chaîne résultant.

Cipher est une des constantes MCRYPT_ciphername

Key est la clé fournie à l'algorithme. Elle doit être tenue secrète.

Data sont les données à traiter.

Mode vaut MCRYPT_ENCRYPT ou MCRYPT_DECRYPT.

IV est le vecteur d'initialisation (optionnel).

Voir aussi: [mdecrypt_cbc\(\)](#), [mdecrypt_cfb\(\)](#), et [mdecrypt_ofb\(\)](#).

[10.39.8 mdecrypt_ofb](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mdecrypt_ofb](#) (int *cipher*, string *key*, string *data*, int *mode*, string *iv*)

string [mdecrypt_ofb](#) (string *cipher*, string *key*, string *data*, int *mode*, string *iv*)

La première syntaxe utilise libmcrypt 2.2.x, et la seconde utilise libmcrypt 2.4.x.

[mdecrypt_ofb\(\)](#) encrypte ou décrypte (suivant le *mode* sélectionné) les données *data* avec le chiffrement *cipher* et la clé *key* en mode OFB et retourne la chaîne résultant.

Cipher est une des constantes MCRYPT_ciphername

Key est la clé fournie à l'algorithme. Elle doit être tenue secrète.

Data sont les données à traiter.

Mode vaut MCRYPT_ENCRYPT ou MCRYPT_DECRYPT.

IV est le vecteur d'initialisation (optionnel).

Voir aussi: [mdecrypt_cbc\(\)](#), [mdecrypt_cfb\(\)](#), et [mdecrypt_ecb\(\)](#).

[10.39.9 mdecrypt_list_algorithms](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mdecrypt_list_algorithms](#) (string *lib_dir*)

[mdecrypt_list_algorithms\(\)](#) sert à lister tous les algorithmes de chiffrement de *lib_dir*.

[mdecrypt_list_algorithms\(\)](#) prend un argument optionnel, qui spécifie le dossier qui contient tous les algorithmes. Si il est omis, la valeur de mdecrypt.algorithms_dir dans php.ini est utilisée.

Exemple avec [mdecrypt_list_algorithms\(\)](#)

```
<?php
$algorithms = mdecrypt_list_algorithms ("/usr/local/lib/libmcrypt");
foreach ($algorithms as $cipher) {
echo $cipher."/n";
}
?>
```

L'exemple ci dessus va affiche tous les algorithmes supportés dans le dossier "/usr/local/lib/libmcrypt".

10.39.10 mcrypt_list_modes

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mcrypt_list_modes](#) (string *lib_dir*)

[mcrypt_list_algorithms\(\)](#) sert à lister tous les modes de chiffrement de *lib_dir*.

[mcrypt_list_algorithms\(\)](#) prend un argument optionnel, qui spécifie le dossier qui contient tous les algorithmes. Si il est omis, la valeur de mcrypt.algorithms_dir dans php.ini est utilisée.

Exemple avec [mcrypt_list_modes\(\)](#)

```
<?php
$modes = mcrypt_list_modes ();
foreach ($modes as $mode) {
echo $mode."/n";
}
?>
```

L'exemple ci dessus va affiche tous les modes supportés dans le dossier "/usr/local/lib/libmcrypt".

10.39.11 mcrypt_get_iv_size

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_get_iv_size](#) (string *cipher*, string *mode*)

[mcrypt_get_iv_size\(\)](#) retourne la taille du Vecteur d'initialisation (VI). En cas d'erreur, la fonction retourne FALSE. Si le VI est ignoré dans le couple chiffrement/mode demandé, zéro est retourné.

Cipher est une constante MCRYPT_ciphername qui indique le nom de l'algorithme sous forme de chaîne.

Mode est une constante MCRYPT_MODE_modename qui peut valoir : "ecb", "cbc", "cfb", "ofb", "nofb" ou "stream".

10.39.12 mcrypt_encrypt

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_encrypt](#) (string *cipher*, string *key*, string *data*, string *mode*, string *iv*)

[mcrypt_encrypt\(\)](#) encrypte les données, et retourne les données cryptées.

Cipher est une constante MCRYPT_ciphername qui indique le nom de l'algorithme sous forme de chaîne.

Key est la clé utilisée pour encrypter les données. Si elle est plus petite que nécessaire, elle sera complétée avec des '\0'.

Data sont les données qui doivent être encryptées. Si la taille des données n'est pas de la forme n * taille_de_bloc, elles seront complétées avec des '\0'. La valeur retournée peut être plus grande que la valeur d'origine.

Mode est une constante MCRYPT_MODE_modename qui peut valoir : "ecb", "cbc", "cfb", "ofb", "nofb" ou "stream".

IV (Vecteur d'initialisation) est utilisé pour les modes CBC, CFB, OFB, et dans certains algorithmes de mode STREAM. Si vous le fournissez par le VI, alors qu'il est nécessaire, la fonction affichera une alerte, et utilise un VI composé de caractères '\0'.

Exemple avec [mcrypt_encrypt\(\)](#)

```
<?php
$iv = mcrypt_create_iv (mcrypt_get_iv_size (MCRYPT_RIJNDAEL_256, MCRYPT_MODE_ECB), MCRYPT_RAND);
$key = "Ceci est une clé secrète";
$text = "Rencontrons nous à 11 heures, derrière le monument";
echo strlen ($text)."\n";
$crypttext = mcrypt_encrypt (MCRYPT_RIJNDAEL_256, $key, $text, MCRYPT_MODE_ECB, $iv);
echo strlen ($crypttext)."\n";
?>
```

L'exemple ci dessus affichera :

```
42
64
```

[10.39.13 mcrypt_decrypt](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_decrypt](#)(string *cipher*, string *key*, string *data*, string *mode*, string *iv*)

Cipher est une constante MCRYPT_ciphername qui indique le nom de l'algorithme sous forme de chaîne. **Key** est la clé utilisée pour encrypter les données. Si elle est plus petite que nécessaire, elle sera complétée avec des '\0'.

Data sont les données qui doivent être encryptées. Si la taille des données n'est pas de la forme n * taille_de_bloc, elles seront complétées avec des '\0'. La valeur retournée peut être plus grande que la valeur d'origine.

Mode est une constante MCRYPT_MODE_modename qui peut valoir : "ecb", "cbc", "cfb", "ofb", "nofb" ou "stream".

IV (Vecteur d'initialisation) est utilisé pour les modes CBC, CFB, OFB, et dans certains algorithmes de mode STREAM. Si vous le fournissez par le VI, alors qu'il est nécessaire, la fonction affichera une alerte, et utilise un VI composé de caractères '\0'.

[10.39.14 mcrypt_module_open](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [mcrypt_module_open](#)(string *algorithm*, string *algorithm_directory*, string *mode*, string *mode_directory*)

[mcrypt_module_open\(\)](#) ouvre le module de l'algorithme et du mode à utiliser. Le nom de l'algorithme est spécifié par le paramètre *algorithm* (par exemple : "twofish"), ou bien une des constantes MCRYPT_ciphername. La librairie est refermée en appelant [@xref{ fonction.mcrypt-module-close , , mcrypt_module_close\(\) }](#), mais il n'est pas nécessaire d'appeler cette fonction si [mcrypt_generic_end\(\)](#) est utilisé. Normalement, [mcrypt_module_open\(\)](#) retourne un pointeur d'encryption, ou bien FALSE en cas d'erreur.

algorithm_directory et *mode_directory* servent à repérer les modules d'encryption. Si vous fournissez un nom de dossier, il sera utilisé. Si vous passez une chaîne vide (""), la valeur utilisé par

mcrypt.algorithms_dir ou *mcrypt.modes_dir* sera celle indiquée dans les directives de configuration. Lorsque ces paramètres ne sont pas fournis les valeurs par défaut, compilées avec la librairie sont utilisées. (généralement /usr/local/lib/libmcrypt).

Exemple avec [mcrypt_module_open\(\)](#)

```
<?php
$td = mcrypt_module_open (MCRYPT_DES, "", MCRYPT_MODE_ECB, "/usr/lib/mcrypt-modes");
?>
```

L'exemple ci dessus va essayer d'ouvrir le module de chiffrement par DES, dans le dossier par défaut, et le mode EBC dans le dossier /usr/lib/mcrypt-modes.

[10.39.15 mcrypt_generic_init](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_generic_init](#)(resource *td*, string *key*, string *iv*)

La taille maximale de la clé doit être celle retournée par [mcrypt_enc_get_key_size\(\)](#) et toutes les valeurs inférieures seront aussi valides. Le vecteur d'initialisation (VI) doit avoir la taille d'un bloc, mais vous devez lire sa taille en appelant [mcrypt_enc_get_iv_size\(\)](#). IV est ignoré en mode ECB. IV DOIT exister en modes CFB, CBC, STREAM, nOFB et OFB. Il doit être aléatoire et unique (mais pas secret). Le même VI doit être utilisé pour le cryptage et le décryptage. Si vous ne voulez pas l'utiliser, remplissez le de zéros, mais ce n'est pas recommandé. La fonction retourne (-1) en cas d'erreur.

Vous devez appeler [mcrypt_generic_init\(\)](#) avant chaque appel à [mcrypt_generic\(\)](#) ou [mdencrypt_generic\(\)](#).

[10.39.16 mcrypt_generic](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_generic](#)(resource *td*, string *data*)

[mcrypt_generic\(\)](#) crypte des données. Les données sont complétées par des "\0" pour obtenir une taille de n fois la taille d'un bloc. Elle retourne les données encryptées. Notez que la longueur de la chaîne retournée peut être plus longue que celle passée en argument, à cause du complément.

[10.39.17 mdencrypt_generic](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mdencrypt_generic](#)(resource *td*, string *data*)

[mdencrypt_generic\(\)](#). Notez que la longueur de la chaîne décryptée peut être plus longue que la chaîne originale, car elle peut avoir été complétée par des "\0".

Exemple avec [mdencrypt_generic\(\)](#)

```
<?php
$iv_size = mcrypt_enc_get_iv_size ($td);
$iv = @mcrypt_create_iv ($iv_size, MCRYPT_RAND);
```

```

if (@mdecrypt_generic_init ($td, $key, $iv) != -1)
{
    $c_t = mcrypt_generic ($td, $plain_text);
@mdecrypt_generic_init ($td, $key, $iv);
    $p_t = mdecrypt_generic ($td, $c_t);
}
if (strcmp ($p_t, $plain_text, strlen($plain_text)) == 0)
echo "ok";
else
echo "erreur";
?>

```

L'exemple ci dessus montre comment vérifier que les données avant cryptage sont bien les mêmes que celles après cryptage/décryptage.

[10.39.18 mcrypt_generic_end](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [mcrypt_generic_end](#)(resource *td*)

[mcrypt_generic_end\(\)](#) termine le cryptage désigné par le pointeur *td*. En fait, elle supprime tous les buffers, et ferme les modules utilisés. Elle retourne FALSE en cas d'erreur, et TRUE sinon.

[10.39.19 mcrypt_enc_self_test](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_enc_self_test](#)(resource *td*)

[mcrypt_enc_self_test\(\)](#) effectue un test du module ouvert et désigné par *td*. Si le test est concluant, elle retourne 0, sinon, 1.

[10.39.20 mcrypt_enc_is_block_algorithm_mode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_enc_is_block_algorithm_mode](#)(resource *td*)

[mcrypt_enc_is_block_algorithm_mode\(\)](#) retourne 1 si ce mode utilise des algorithmes par blocs, et 0 sinon. (i.e. 0 pour stream, et 1 pour cbc, cfb, ofb).

[10.39.21 mcrypt_enc_is_block_algorithm](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_enc_is_block_algorithm](#)(resource *td*)

[mcrypt_enc_is_block_algorithm\(\)](#) retourne 1 si l'algorithme utilisé est un algorithme par bloc, et 0 si c'est un algorithme par flot.

[10.39.22 mcrypt enc is block mode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_enc_is_block_mode](#)(resource *td*)

[mcrypt_enc_is_block_mode\(\)](#) retourne 1 si le mode retourne des blocs d'octets, ou bien 0 si il retourne des octets (par flot). (i.e. 1 pour cbc et ecb, et 0 pour cfb et stream).

[10.39.23 mcrypt enc get block size](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_enc_get_block_size](#)(resource *td*)

[mcrypt_enc_get_block_size\(\)](#) retourne la taille de blocs d'un algorithme en octets.

[10.39.24 mcrypt enc get key size](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_enc_get_key_size](#)(resource *td*)

[mcrypt_enc_get_key_size\(\)](#) retourne la taille maximale de clé acceptée par le mode désigné par *td*, en octets.

[10.39.25 mcrypt enc get supported key sizes](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mcrypt_enc_get_supported_key_sizes](#)(resource *td*)

[mcrypt_enc_get_supported_key_sizes\(\)](#) retourne un tableau contenant les tailles des clés supportées par l'algorithme désigné par *td*. Si il retourne un tableau vide, c'est que toutes les clés entre 1 et [mcrypt_enc_get_key_size\(\)](#) sont acceptées par l'algorithme.

[10.39.26 mcrypt enc get iv size](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_enc_get_iv_size](#)(resource *td*)

[mcrypt_enc_get_iv_size\(\)](#) retourne la taille du VI de l'algorithme désigné par *td*, en octets. Si la valeur retournée est 0, c'est que l'algorithme ne demande pas de VI. Un VI est demandé en mode cbc, cfb et ofb, et parfois en mode stream.

[10.39.27 mcrypt enc get algorithms name](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_enc_get_algorithms_name](#)(resource *td*)

[mcrypt_enc_get_algorithms_name\(\)](#) retourne le nom de l'algorithme désigné par *td*.

[10.39.28 mcrypt_enc_get_modes_name](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_enc_get_modes_name](#)(resource *td*)

[mcrypt_enc_get_modes_name\(\)](#) retourne le nom du mode désigné par *td*.

[10.39.29 mcrypt_module_self_test](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [mcrypt_module_self_test](#)(string *algorithm*, string *lib_dir*)

[mcrypt_module_self_test\(\)](#) effectue un test sur l'algorithme spécifié. Le paramètre optionnel *lib_dir* contient le chemin jusqu'au module de l'algorithme sur le système.

Retourne TRUE si le test fonctionne, et FALSE sinon.

[10.39.30 mcrypt_module_is_block_algorithm_mode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [mcrypt_module_is_block_algorithm_mode](#)(string *mode*, string *lib_dir*)

[mcrypt_module_is_block_algorithm_mode\(\)](#) retourne TRUE si le mode doit être utilisé avec un algorithme par bloc, sinon retourne 0 (i.e. 0 pour stream, et 1 pour cbc, cfb, ofb). Le paramètre optionnel *lib_dir* contient le chemin jusqu'au module de l'algorithme sur le système.

[10.39.31 mcrypt_module_is_block_algorithm](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [mcrypt_module_is_block_algorithm](#)(string *algorithm*, string *lib_dir*)

[mcrypt_module_is_block_algorithm\(\)](#) retourne TRUE si *algorithm* est un algorithme par bloc, sinon retourne 0. Le paramètre optionnel *lib_dir* contient le chemin jusqu'au module de l'algorithme sur le système.

[10.39.32 mcrypt_module_is_block_mode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [mcrypt_module_is_block_mode](#)(string *mode*, string *lib_dir*)

[mcrypt_module_is_block_mode\(\)](#) retourne TRUE si ce mode fournit des blocs d'octets, ou bien un flot d'octets. (i.e. 1 pour cbc et ecb, et 0 pour cfb et stream). Le paramètre optionnel *lib_dir* contient le chemin jusqu'au module de l'algorithme sur le système.

[10.39.33 mcrypt_module_get_algo_block_size](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_module_get_algo_block_size](#)(string *algorithm*, string *lib_dir*)

[mdecrypt_module_get_algo_block_size\(\)](#) retourne la taille de bloc d'un algorithme, en octets. Le paramètre optionnel *lib_dir* contient le chemin jusqu'au module de l'algorithme sur le système.

[10.39.34 mdecrypt_module_get_algo_key_size](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mdecrypt_module_get_algo_key_size](#)(string *algorithm*, string *lib_dir*)

[mdecrypt_module_get_algo_key_size\(\)](#) retourne la taille maximale de la clé supporté par l'algorithme *algorithm*. Le paramètre optionnel *lib_dir* contient le chemin jusqu'au module de l'algorithme sur le système.

[10.39.35 mdecrypt_module_get_algo_supported_key_sizes](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array @xref{function.mdecrypt-module-enc-get-algo-supported-key-sizes , , mdecrypt_module_enc_get_algo_supported_key_sizes (string *algorithm*, string *lib_dir*) }

@xref{function.mdecrypt-module-enc-get-algo-supported-key-sizes , , mdecrypt_module_enc_get_algo_supported_key_sizes()} retourne un tableau avec les tailles des clés supportées par l'algorithme *algorithm*. Si le tableau retourné est vide, c'est que toutes les tailles de clé entre 1 et [mdecrypt_module_get_algo_key_size\(\)](#) sont supportées. Le paramètre optionnel *lib_dir* peut contenir le dossier du module sur le système.

[10.40 Hash](#)

[\[Notes en ligne\]](#)

Ces fonctions ont été prévues pour fonctionner avec [mhash](#).

Cet ensemble de fonctions représente une interface avec la librairie mhash. mhash accepte un grand nombre d'algorithmes différents, tels que MD5, SHA1, GOST, bien d'autres.

Pour l'utiliser, téléchargez les distributions de mhash depuis le site [web ici](#) et suivez les instructions d'installation incluses. Vous aurez besoin de recompiler PHP avec l'option `--with-mhash` pour activer cette extension.

mhash sert à calculer des sommes de vérifications, des signatures de messages, etc...

. Calcule un hash de type SHA1 et l'affiche au format hexadécimal

```
<?php
$input = "Rencontrons nous à 9h00 dans notre repaire secret.";
$hash = mhash(MHASH_SHA1, $input);
print "Le hash est ".bin2hex($hash)."\n";
?>
```

Cela va produire quelque chose du type (Note du Traducteur : c'est le hash de la version anglaise)

```
Le hash est d3b85d710d8f6e4e5efd4d5e67d041f9cecedafe
```

Pour avoir une liste complète des hash supportés, reportez vous à la documentation de mhash. En règle générale, vous pouvez utiliser un algorithme de hash avec le type : MHASH_NOMDEHASH. Par exemple

pour utiliser HAVAL vous devez spécifier la constante PHP MHASH_HAVAL.

Voici une liste de hash qui sont actuellement supportés par mhash. Si un hash n'est pas dans la liste, mais qu'il est disponible avec mhash, c'est que ce document a pris de l'âge.

- MHASH_MD5
- MHASH_SHA1
- MHASH_HAVAL
- MHASH_RIPEMD160
- MHASH_RIPEMD128
- MHASH_SNEFRU
- MHASH_TIGER
- MHASH_GOST
- MHASH_CRC32
- MHASH_CRC32B

[10.40.1 mhash_get_hash_name](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mhash_get_hash_name](#)(int *hash*)

[mhash_get_hash_name\(\)](#) sert à connaître le nom d'un hash.

[mhash_get_hash_name\(\)](#) prend un numéro d'identifiant de hash, et retourne son nom, ou bien FALSE si le hash n'existe pas, ou si une erreur est survenue.

exemple mhash_get_hash_name

```
<?php
$hash = MHASH_MD5;
print mhash_get_hash_name($hash);
?>
```

L'exemple ci dessus va afficher :

MD5

[10.40.2 mhash_get_block_size](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mhash_get_block_size](#)(int *hash*)

[mhash_get_block_size\(\)](#) sert à connaître la taille de bloc du hash spécifié *hash*.

[mhash_get_block_size\(\)](#) prend un seul argument : le *hash* et retourne la taille en octets, ou bien FALSE si le *hash* n'existe pas.

10.40.3 mhash_count

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mhash_count](#)(void)

[mhash_count\(\)](#) retourne l'identifiant de hash maximal. Les hash sont numérotés de 0 jusqu'à cet identifiant.

Parcourir la liste des hash

```

<?php
$nr = mhash_count();
for($i = 0; $i <= $nr; $i++) {
echo sprintf("The blocksize of %s is %d\n",
mhash_get_hash_name($i),
mhash_get_block_size($i));
}
?>

```

10.40.4 mhash

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mhash](#)(int *hash*, string *data*)

[mhash\(\)](#) applique la fonction de hash *hash* aux données *data* et retourne le résultat.

10.40.5 mhash_keygen_s2k

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mhash_keygen_s2k](#)(int *hash*, string *password*, string *salt*, int *bytes*)

[mhash_keygen_s2k\(\)](#) génère une clé de *bytes* octets de long, à partir d'un mot de passe. Cette fonction utilise l'algorithme Salted S2K, spécifié dans OpenPGP (RFC 2440). Cet algorithme va utiliser l'algorithme de hashage *hash* pour créer la clé. Le paramètre *salt* doit être différent et suffisamment aléatoire pour chaque clé que vous générez, afin de créer des clés différentes. Ce grain de sel reservira lorsque vous vérifierez les clés : c'est alors une bonne idée que de l'ajouter à la fin de la clé générée. *salt* doit avoir la longueur de 8 octets, et sera complété par des 0 si vous ne fournissez pas suffisamment de données. N'oubliez pas que les mots de passe fournis par les utilisateurs ne sont pas conseillé pour faire des clés cryptographique, étant donné que les utilisateurs normaux retiennent des mots de passe qu'ils peuvent saisir au clavier. Ces mots de passe utilisent uniquement 6 à 7 des 8 bits d'un caractère (voir moins). Il est vivement recommandé d'appliquer une fonction de transformation (comme celle ci), à un mot de passe utilisateur.

10.41 Fonctions diverses

[\[Notes en ligne\]](#)

Ces fonctions ont été placées là, car elles ne rentraient dans aucune catégorie adéquate.

[10.41.1 connection_aborted](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [connection_aborted](#) (void)

Retourne TRUE si le client a abandonné la connexion. Reportez vous à [8.1 Gestion des connexions](#) du chapitre [8 Caractéristiques](#).

[10.41.2 connection_status](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [connection_status](#) (void)

Retourne les bits de status de la connexion. Reportez vous à la section [8.1 Gestion des connexions](#) pour plus de détails.

[10.41.3 connection_timeout](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [connection_timeout](#) (void)

Retourne TRUE si le script a expiré. Reportez vous à la section [8.1 Gestion des connexions](#) pour plus de détails.

[10.41.4 define](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [define](#) (string *name*, mixed *value*, int *case_insensitive*)

Définit une constante, de la même façon qu'une variable, sauf que :

- Les constantes ne commencent pas par le signe '\$'
- Les constantes sont accessibles partout, de manière globale.
- Les constantes ne peuvent pas être redéfinies, ou indéfinies, une fois qu'elles ont été définies.
- Les constantes ne représentent que des valeurs scalaires : il n'est pas possible de définir des tableaux ou des objets.

Le nom de la constante est donnée par le paramètre *name*; sa valeur est donnée par *value*.

Le troisième paramètre optionnel *case_insensitive* peut prendre la valeur de *1*. Dans ce cas, le nom de la constante sera insensible à la casse (c'est la valeur par défaut). Cela signifie que, par défaut, CONSTANT et Constant représentent des valeurs différentes.

Définition d'une constante

```
<?php
define ("CONSTANT", "Bonjour le monde.");
```

```
echo CONSTANT; // affiche "Bonjour le monde."
?>
```

[define\(\)](#) retourne TRUE en cas de succès et FALSE sinon.
Voir aussi [defined\(\)](#) et la section sur les [9.2 Les constantes](#).

[10.41.5 defined](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [defined](#)(string *name*)

Retourne TRUE si la constante nommée *name* a été définie, et FALSE sinon.
Voir aussi [define\(\)](#) et la section sur les [9.2 Les constantes](#).

[10.41.6 die](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [die](#)(string *message*)

Cette fonction affiche la chaîne passée en paramètre, puis termine l'exécution du script. Il ne retourne rien de plus.

Exemple avec [die\(\)](#)

```
<?php
$filename = '/path/to/data-file';
$file = fopen ($filename, 'r')
or die("impossible d'ouvrir le fichier ($filename)");
?>
```

[10.41.7 eval](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [eval](#)(string *code_str*)

[eval\(\)](#) évalue la chaîne *code_str* comme un script PHP. Parmi les utilisations possibles, cette fonction permet de stocker du code dans une base de données, pour utilisation ultérieure.

Il faut bien garder en tête que le code passé à [eval\(\)](#) doit être valide, y compris les points virgules de fin de ligne, et les séquences d'échappement, sinon l'exécution se terminera.

N'oubliez pas que les variables utilisées dans la fonction [eval\(\)](#) resteront accessibles dans le script principal.
exemple avec [eval\(\)](#) – inclusion de texte *exemple avec [eval\(\)](#) – inclusion de texte*

```
<?php
$string = 'tasse';
```

```

$name = 'cafe';
$string = 'Ceci est une $string avec mon $name dedans.<br>';
echo $string;
eval( "\$string = \"\$string\";\" );
echo $string;
?>

```

L'exemple ci dessus devrait afficher :

```

Ceci est une $string avec mon $name dedans.
Ceci est une tasse avec mon cafe dedans.

```

10.41.8 exit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [exit](#)

Cette fonction termine l'analyse d'un script en cours d'exécution. Elle ne renvoie aucune valeur.

10.41.9 get_browser

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [get_browser](#)(string *user_agent*)

[get_browser\(\)](#) essaie de déterminer les capacités du navigateur client. Cela se fait en lisant les informations dans le fichier ``browscap.ini'`. Par défaut, la valeur de `$HTTP_USER_AGENT` est utilisée. Cependant, vous pouvez passer n'importe quelle valeur avec le paramètre optionnel *user_agent* à [get_browser\(\)](#). Les informations sont retournées sous forme d'un objet, dont les différents membres contiendront des informations, telles que les versions majeures et mineures, et des chaînes d'identification; des booléens pour des caractéristiques telles que frames, JavaScript, and cookies; et ainsi de suite.

Même si ``browscap.ini'` contient des informations sur de nombreux clients, il compte sur les utilisateurs pour être mis à jour. Le format du fichier est facilement compréhensible.

L'exemple suivant montre comment on peut lister les informations disponibles : *Exemple avec [get_browser\(\)](#)*

```

<?php
function list_array ($array) {
while (list ($key, $value) = each ($array)) {
    $string .= "<b>$key:</b> $value<br>\n";
}
return $string;
}
echo "$HTTP_USER_AGENT<hr>\n";
$browser = get_browser();
echo list_array ((array) $browser);
?>

```

L'affichage devrait ressembler à ceci :

```

Mozilla/4.5 [en] (X11; U; Linux 2.2.9 i586)<hr>
<b>browser_name_pattern:</b> Mozilla/4\..*<br>
<b>parent:</b> Netscape 4.0<br>
<b>platform:</b> Unknown<br>
<b>majorver:</b> 4<br>
<b>minorver:</b> 5<br>
<b>browser:</b> Netscape<br>
<b>version:</b> 4<br>
<b>frames:</b> 1<br>
<b>tables:</b> 1<br>
<b>cookies:</b> 1<br>
<b>backgroundsounds:</b> <br>
<b>vbscript:</b> <br>
<b>javascript:</b> 1<br>
<b>javaapplets:</b> 1<br>
<b>activexcontrols:</b> <br>
<b>beta:</b> <br>
<b>crawler:</b> <br>
<b>authenticcodeupdate:</b> <br>
<b>msn:</b> <br>

```

Pour fonctionner, votre configuration [7.1.13 Directives de configuration du navigateur](#) doit mener au fichier `browscap.ini`.

Pour plus d'informations, (y compris pour les endroits où charger le fichier `browscap.ini`), suivez la FAQ PHP à <http://www.php.net/FAQ.html>.

Note : *Browscap a été ajouté dans PHP version 3.0b2.*

[10.41.10 ignore_user_abort](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ignore_user_abort](#)(int *setting*)

Cette fonction active l'option décidant si, lors de la déconnexion du client, le script doit poursuivre son exécution ou non. La fonction renvoie le paramétrage précédent et elle peut être appelée sans argument pour ne pas changer le paramétrage courant. Voir le paragraphe gestion des connexions dans le chapitre caractéristiques pour une description plus complète des manipulations de connexion en PHP.

[10.41.11 iptcparsed](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [iptcparsed](#)(string *iptcblock*)

Analyse un bloc binaire IPTC et recherche les balises simples. Elle retourne un tableau avec les balises comme index, et les valeurs dans les valeurs de tableau correspondantes. En cas d'erreur, ou si aucune balise IPTC n'a été trouvée, retourne FALSE. Voir [getimagesize\(\)](#) pour un exemple.

[10.41.12 leak](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [leak](#)(int *bytes*)

[leak\(\)](#) crée une fuite de mémoire.

Cette fonction est pratique pour débogger le gestionnaire de mémoire, qui doit nettoyer automatiquement les fuites de mémoire après chaque requête.

[10.41.13 pack](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pack](#) (string *format*, mixed *args* ...)

Compacte les arguments dans une chaîne binaire, suivant le format *format*. Retourne la chaîne binaire. L'idée vient du Perl, et tout le formatage fonctionne de la même fonction qu'en Perl, mais quelques formats manquent encore (comme, "u"). La chaîne de format est composée d'une série de code de formats, suivis un quantificateur optionnel. Le quantificateur peut être un entier, ou * pour la répétition indéfinie. Pour a, A, h et H le quantificateur spécifie combien de caractères d'un argument sont pris; pour @, c'est la position absolue où placer les données, et pour le reste, c'est le nombre de répétitions. Actuellement, les formats suivants sont implémentés :

- une chaîne complétée avec NUL
- une chaîne complétée avec espace (SPACE)
- Chaîne hexadécimale h, bit de poids faible en premier.
- Chaîne hexadécimale H, bit de poids fort en premier.
- c caractère signé
- C caractère non signé
- s entier court signé (toujours sur 16 bits, ordre des bits dépendant de la machine).
- S entier court non signé (toujours 16 bits, ordre des bits dépendant de la machine).
- n entier court signé (toujours 16 bits, ordre des bits big endian)
- v entier courts non signé (toujours 16 bits, ordre des bits little endian)
- i entier signé (taille et ordre des bits dépendants de la machine)
- I entier non signé (taille et ordre des bits dépendants de la machine)
- l entier long signé (toujours 32 bits, ordre des bits dépendant de la machine)
- L entier long non signé (always 32 bits, ordre des bits dépendant de la machine)
- N entier long non signé (toujours 16 bits, ordre des bits big endian)
- V ntier long non signé (toujours 16 bits, ordre des bits little endian)
- f nombre à virgule flottante (taille et représentation dépendantes de la machine)
- d nombre à virgule flottante double (taille et représentation dépendantes de la machine)
- x bit NUL
- X recule d'un octet
- rempli avec NUL, jusqu'à une position absolue

Compactage d'une chaîne

```
$binarydata = pack ("nvc*", 0x1234, 0x5678, 65, 66);
```

La chaîne binaire résultante aura 6 octets de long, et contiendra la séquence 0x12, 0x34, 0x78, 0x56, 0x41, 0x42.

Notez que la distinction entre signé et non signé n'affecte que la fonction [unpack\(\)](#), tandis que la fonction [pack\(\)](#) fournira le même résultat pour les deux formats.

De plus, notez que PHP enregistre de manière interne et intégrale les valeurs : cette représentation dépend de la machine. Si vous essayez d'enregistrer une valeur trop grande, elle risque d'être convertie, et de donner lieu à des effets de bords vicieux.

[10.41.14 serialize](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [serialize](#) (mixed *value*)

[serialize\(\)](#) retourne une chaîne contenant un flot d'octets représentant la valeur *value*, et qui pourra être relue. Ceci est très utile pour enregistrer ou passer des valeurs à PHP sans perdre leur type ou leur structure. Pour relire la chaîne dans PHP, utilisez [unserialize\(\)](#). [serialize\(\)](#) accepte les integer, double, string, array (multidimensionnel) et object (les propriétés de l'objet seront sauvegardées, mais pas ses méthodes).

Exemple avec [serialize\(\)](#)

```
// $session_data contient un tableau multi-dimensionnel avec des informations
// de session concernant l'utilisateur courant. On utilise serialize() pour
// enregistrer les informations dans une base de donnée à la fin de la requête.
$conn = odbc_connect("webdb", "php", "poulet");
$stmt = odbc_prepare($conn,
                    "UPDATE sessions SET data = ? WHERE id = ?");
$sqldata = array(serialize($session_data), $PHP_AUTH_USER);
if (!odbc_execute($stmt, 38;$sqldata)) {
    $stmt = odbc_prepare($conn,
                        "INSERT INTO sessions (id, data) VALUES(?, ?)");
if (!odbc_execute($stmt, 38;$sqldata)) {
    /* Un os ! Souffre et potasse. */
}
}
```

[10.41.15 sleep](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [sleep](#) (int *seconds*)

La fonction sleep retarde l'exécution du programme pendant *seconds* secondes.

Voir aussi [usleep\(\)](#).

[10.41.16 uniqid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [uniqid](#) (string *prefix*, boolean *lcg*)

[uniqid\(\)](#) retourne un identifiant préfixé unique, basé sur l'heure courante, en micro-secondes. Le préfixe peut servir à identifier facilement différents hôtes, si vous générez simultanément des fichiers depuis plusieurs hôtes, à la même micro-seconde. *prefix* peut prendre jusqu'à 114 caractères.

Si le paramètre optionnel *lcg* est TRUE, [uniqid\(\)](#) ajoutera une entropie "combined LCG" à la fin de la valeur

retournée, ce qui renforcera encore l'unicité de l'identifiant.

Sans *prefix* (préfixe vide), la chaîne retournée fera 13 caractères. Si *lcg* est à TRUE, elle fera 23 caractères.

Note : *Le paramètre lcg est utilisé à partir de PHP 4 et PHP 3.0.13 et ultérieurs.*

Si vous voulez utiliser un identifiant unique, ou bien gérer des cookies, il est recommandé d'utiliser un code tel que celui ci :

```
$token = md5 (uniqid ("")); // pas de section aléatoire.
$better_token = md5 (uniqid (rand())); // mieux, difficile à deviner
```

Ceci va créer un identifiant de 32 caractère (un nombre hexadécimal de 128) qui sera très difficile à prédire.

10.41.17 unpack

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [unpack](#) (string *format*, string *data*)

[unpack\(\)](#) déconditionne des données depuis une chaîne binaire avec le format *format*. Retourne un tableau contenant les éléments déconditionnés.

[unpack\(\)](#) se comporte légèrement différemment de la version Perl car les données déconditionnées sont stockées dans un tableau. Pour cela, il faut donner un nom à chaque format utilisé, et les séparer par des slash (/). *Exemple avec [unpack\(\)](#)*

```
$array = unpack ("c2chars/nint", $binarydata);
```

Le tableau résultant contiendra les entrées suivantes : "chars1", "chars2" et "int".

Pour plus de détails, reportez vous à: [pack\(\)](#)

Il faut noter que PHP gère les valeurs en interne sous forme signée. Si vous déconditionnez une valeur qui est aussi grande que la taille utilisée en interne par PHP, le résultat se trouvera être un nombre négatif, même si il a été déconditionné avec l'option " non signé ".

10.41.18 unserialize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [unserialize](#) (string *str*)

[unserialize\(\)](#) prend une variable créée avec [serialize\(\)](#) et la convertie en valeur PHP. La valeur lue est retournée, et elle peut être de type integer, double, string, array or object. Si un objet a été sérialisé, ses méthodes ont été perdues, et ses attributs ont été conservés.

Exemple avec [unserialize\(\)](#)

```
// Ici, on utilise unserialize() pour lire les données concernant
// la session d'un utilisateur dans $session_data. Cet exemple complète
// celui décrit dans serialize\(\).
$conn = odbc_connect("webdb", "php", "chicken");
```

```

$stmt = odbc_prepare($conn, "SELECT data FROM sessions WHERE id = ?");
$sqldata = array($PHP_AUTH_USER);
if (!odbc_execute($stmt, 38:$sqldata) || !odbc_fetch_into($stmt, 38:$tmp)) {
    // En cas d'erreur, initialisation à tableau vide.
    $session_data = array();
} else {
    // On a maintenant les information dans $tmp[0].
    $session_data = unserialize($tmp[0]);
if (!is_array($session_data)) {
    // Un os ! initialisation à tableau vide
    $session_data = array();
}
}
}

```

10.41.19 usleep

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [usleep](#)(int *micro_seconds*)

La [sleep\(\)](#) retarde l'exécution du programme pendant *micro_seconds* micro-secondes.

Voir aussi [sleep\(\)](#).

Note : *Cette fonction est inopérante sous Windows*

10.42 fonctions mSQL

[\[Notes en ligne\]](#)

10.42.1 msql

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [msql](#)(string *database*, string *query*, int *link_identifier*)

Retourne un identifiant positif de résultat de requête, ou FALSE en cas d'erreur.

[msql\(\)](#) sélectionne la base de données *database*, et y exécute la requête. Si l'identifiant de connexion n'est pas fourni, la fonction va rechercher un lien ouvert à un serveur mSQL, et sinon, il va tenter d'en créer une, avec [msql_connect\(\)](#), sans argument.

10.42.2 msql_affected_rows

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [msql_affected_rows](#)(int *query_identifier*)

Retourne le nombre de lignes affectées par la dernière commande INSERT, UPDATE ou DELETE sur le serveur associé au *link_identifier*. Si ce dernier n'est pas précisé, la dernière connexion est utilisée.

Voir aussi: [msql_query\(\)](#).

10.42.3 [mysql_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_close](#) (int *link_identifiant*)

Retourne TRUE en cas de succès, FALSE en cas d'erreur.

[mysql_close\(\)](#) ferme la connexion au serveur de base de données MySQL référencé par l'identifiant fourni. Si aucun identifiant n'est fourni, la dernière connexion sera utilisée.

Notez bien qu'il n'est pas toujours nécessaire d'appeler cette fonction, car les connexions non persistantes seront automatiquement fermées à la fin du script.

[mysql_close\(\)](#) ne peut pas fermer les connexions persistantes, générées par [mysql_pconnect\(\)](#).

Voir aussi: [mysql_connect\(\)](#) et [mysql_pconnect\(\)](#).

10.42.4 [mysql_connect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_connect](#) (string *hostname*)

Retourne un identifiant de connexion positif en cas de succès, et FALSE sinon.

[mysql_connect\(\)](#) établit une connexion à un serveur MySQL. Le nom d'hôte est optionnel, et lorsqu'il manque, localhost est utilisé.

Si un deuxième appel est fait à [mysql_connect\(\)](#), avec les mêmes arguments, ce ne sera pas une nouvelle connexion qui va être ouverte, mais l'ancienne connexion qui sera utilisée, et son identifiant sera retourné.

Le lien au serveur sera fermé dès la fin du script, ou bien, manuellement, lors de l'appel de [mysql_close\(\)](#).

Voir aussi [mysql_pconnect\(\)](#) et [mysql_close\(\)](#).

10.42.5 [mysql_create_db](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_create_db](#) (string *database name*, int *link_identifiant*)

[mysql_create_db\(\)](#) essaie de créer une nouvelle base de données sur le serveur référencé par l'identifiant *link_identifiant*.

Voir aussi: [mysql_drop_db\(\)](#).

10.42.6 [mysql_createdb](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_createdb](#) (string *database name*, int *link_identifiant*)

Identique à [mysql_create_db\(\)](#).

10.42.7 [mysql_data_seek](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_data_seek](#) (int *query_identifiant*, int *row_number*)

Retourne TRUE en cas de succès, et FALSE en cas d'échec.

[mysql_data_seek\(\)](#) déplace le pointeur interne de résultat mSQL, et le place à l'offset donné. Le prochain appel à la fonction [mysql_fetch_row\(\)](#) retournera cette ligne.

Voir aussi: [mysql_fetch_row\(\)](#).

10.42.8 mysql_dbname

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_dbname](#)(int *query_identifieur*, int *i*)

[mysql_dbname\(\)](#) retourne le nom de la base de données enregistré en position *i* du pointeur de résultat retourné par la fonction [mysql_listdbs\(\)](#). La fonction [mysql_numrows\(\)](#) peut être utilisée pour déterminer le nombre de nom de bases disponibles.

10.42.9 mysql_drop_db

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_drop_db](#)(string *database_name*, int *link_identifieur*)

Retourne TRUE en cas de succès, et FALSE en cas d'échec.

[mysql_drop_db\(\)](#) essaie d'effacer une base de données entière sur le serveur référencé par l'identifiant fourni.

Voir aussi: [mysql_create_db\(\)](#).

10.42.10 mysql_dropdb

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Voir [mysql_drop_db\(\)](#).

10.42.11 mysql_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_error](#)()

Les erreurs générées par mSQL ne sont plus traitées comme des alertes. Au lieu de cela, elles sont stockées, et accessibles à partir de cette fonction.

10.42.12 mysql_fetch_array

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_fetch_array](#)(int *query_identifieur*, int *result_type*)

Retourne un tableau qui contient la ligne demandée, ou FALSE, si il n'y a pas d'autres lignes.

[mysql_fetch_array\(\)](#) est une version évoluée de [mysql_fetch_row\(\)](#). En plus d'enregistrer les données dans un tableau à indice numérique, il peut enregistrer les données dans un tableau associatif, en utilisant les noms des champs comme clés.

Le deuxième argument *result_type* de [mysql_fetch_array\(\)](#) est une constante, et peut prendre les valeurs suivantes : MYSQL_ASSOC, MYSQL_NUM, et MYSQL_BOTH.

Méfiez vous des requêtes qui retournent une ligne qui ne contient qu'un champs de valeur 0 (ou NULL, ou chaîne vide).

Il est important de noter que [mysql_fetch_array\(\)](#) est marginalement plus lent que [mysql_fetch_row\(\)](#), alors qu'elle apporte un confort d'utilisation appréciable.

Voir aussi [mysql_fetch_row\(\)](#).

[10.42.13 mysql_fetch_field](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [mysql_fetch_field](#) (int *query_identifieur*, int *field_offset*)

Retourne un objet contenant les informations sur un champs.

[mysql_fetch_field\(\)](#) sert à lire les informations sur les champs, dans certaines requêtes. Si l'offset du champs n'est pas spécifié, le prochain champs sera retourné.

Les propriétés de l'objet sont :

- name – nom de la colonne
- table – nom de la table à qui appartient la colonne.
- not_null – 1 si la colonne ne peut être NULL
- primary_key – 1 si la colonne est une clé primaire
- unique – 1 la colonne est une clé unique
- type – le type de la colonne

Voir aussi [mysql_field_seek\(\)](#).

[10.42.14 mysql_fetch_object](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_fetch_object](#) (int *query_identifieur*, int *result_type*)

Retourne un objet, dont les propriétés seront affectées suivant les champs de la ligne lue, ou FALSE si il ne reste plus de lignes.

[mysql_fetch_object\(\)](#) est identique à [mysql_fetch_array\(\)](#), avec une différence : c'est un objet qui est retourné, à la place d'un tableau. Par conséquent, cela signifie que vous ne pouvez accéder aux valeurs que par les noms des champs, et non plus avec leur offset. (les nombres sont interdits dans les noms de propriétés)

L'argument optionnel *result_type* de [mysql_fetch_array\(\)](#) est une constante qui peut prendre les valeurs suivantes : MYSQL_ASSOC, MYSQL_NUM, et MYSQL_BOTH.

Cette fonction est aussi rapide que [mysql_fetch_array\(\)](#), et marginalement plus lente que [mysql_fetch_row\(\)](#) (la différence est non significative).

Voir aussi: [mysql_fetch_array\(\)](#) et [mysql_fetch_row\(\)](#).

[10.42.15 mysql_fetch_row](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mysql_fetch_row](#) (int *query_identifieur*)

Retourne un tableau qui contient la ligne demandée, ou FALSE, si il n'y a plus de lignes à lire.

[mysql_fetch_row\(\)](#) retourne une ligne, extraite du résultat associé à l'identifiant de résultat *query_identifieur*. La ligne est retournée sous la forme d'un tableau. Chaque résultat est enregistré dans un champs, indexé numériquement, à partir de 0.

Les appels ultérieurs à [mysql_fetch_row\(\)](#) retourneront les lignes suivantes, ou FALSE, lorsqu'il n'y aura plus de ligne.

Voir aussi: [mysql_fetch_array\(\)](#), [mysql_fetch_object\(\)](#), [mysql_data_seek\(\)](#) et [mysql_result\(\)](#).

10.42.16 mysql_fieldname

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_fieldname](#)(int *query_identifieur*, int *field*)

[mysql_fieldname\(\)](#) retourne le nom du champs à l'index *field*. *query_identifieur* est un identifiant de résultat, et *field* est un index de champs. `mysql_fieldname($result, 2);` retournera le nom du deuxième champs, dans le résultat associé à *query_identifieur*.

10.42.17 mysql_field_seek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_field_seek](#)(int *query_identifieur*, int *field_offset*)

Recherche l'offset du champs *field_offset*. Le prochain appel à [mysql_fetch_field\(\)](#) qui d'aura pas d'argument *field_offset*, retournera ce champs.

Voir aussi: [mysql_fetch_field\(\)](#).

10.42.18 mysql_fieldtable

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_fieldtable](#)(int *query_identifieur*, int *field*)

Retourne le nom de la table d'ou est le champs *field* a été extrait.

10.42.19 mysql_fieldtype

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_fieldtype](#)(int *query_identifieur*, int *i*)

[mysql_fieldtype\(\)](#) est similaire à [mysql_fieldname\(\)](#). Les arguments sont identiques, mais c'est le type du champs qui est retourné. Cela produira un résultat tel que "int", "string" ou "real".

10.42.20 mysql_fieldflags

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_fieldflags](#)(int *query_identifieur*, int *i*)

[mysql_fieldflags\(\)](#) retourne le flag du champs spécifié. Actuellement, il peut valoir soit "not null", "primary key", ou une combinaison des deux ou "" (chaîne vide).

[10.42.21 mysql_fieldlen](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_fieldlen](#)(int *query_identifieur*, int *i*)

[mysql_fieldlen\(\)](#) retourne la longueur d'un champs.

[10.42.22 mysql_free_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_free_result](#)(int *query_identifieur*)

[mysql_free_result\(\)](#) Libère de la mémoire le résultat associé à l'identifiant de résultat *query_identifieur*.

Lorsque PHP a terminé une requête, cette mémoire est libérée, ce qui fait que vous n'aurez pas besoin de cette fonction. Vous pouvez toujours l'utiliser pour vous assurez que vous n'utilisez pas trop de mémoire durant un script.

[10.42.23 mysql_freeresult](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Voir [mysql_free_result\(\)](#).

[10.42.24 mysql_list_fields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_list_fields](#)(string *database*, string *tablename*)

[mysql_list_fields\(\)](#) lit les informations de la table donnée. Les arguments sont le nom de la base de données, et le nom de la table. Cette fonction retourne un identifiant de résultat qui sera utilisé avec [mysql_fieldflags\(\)](#), [mysql_fieldlen\(\)](#), [mysql_fieldname\(\)](#) et [mysql_fieldtype\(\)](#). Un identifiant de résultat est un entier positif. La fonction retourne -1 si une erreur survient. Une chaîne décrivant l'erreur sera placée dans la variable \$phperrormsg, et à moins que cette fonction n'ai été appelée avec (@mysql_list_fields()), alors cette erreur sera affichée.

Voir aussi [mysql_error\(\)](#).

[10.42.25 mysql_listfields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Voir [mysql_list_fields\(\)](#).

[10.42.26 mysql_list_dbs](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_list_dbs](#)

[mysql_list_dbs\(\)](#) retourne un pointeur de résultat, qui contiendra les noms des bases de données disponibles sur la connexion mSQL courante. Utilisez [mysql_dbname\(\)](#) pour passer en revue toutes les lignes.

[10.42.27 mysql_listdbs](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Voir [mysql_list_dbs\(\)](#).

[10.42.28 mysql_list_tables](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_list_tables](#)(string *database*)

[mysql_list_tables\(\)](#) prend un nom de base de données, et fourni un résultat, un peu comme la fonction [mysql\(\)](#). La fonction [mysql_tablename\(\)](#) devrait être utilisée de préférence pour extraire les nom de table d'un pointeur de résultat.

[10.42.29 mysql_listtables](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Voir [mysql_list_tables\(\)](#).

[10.42.30 mysql_num_fields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_num_fields](#)(int *query_identifieur*)

[mysql_num_fields\(\)](#) retourne le nombre de champs du résultat associé à l'identifiant *query_identifieur*. Voir aussi: [mysql\(\)](#), [mysql_query\(\)](#), [mysql_fetch_field\(\)](#) et [mysql_num_rows\(\)](#).

[10.42.31 mysql_num_rows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_num_rows](#)(int *query_identifieur*)

[mysql_num_rows\(\)](#) retourne le nombre de lignes du résultat associé à l'identifiant *query_identifieur*. Voir aussi: [mysql\(\)](#), [mysql_query\(\)](#) et [mysql_fetch_row\(\)](#).

[10.42.32 mysql_numfields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_numfields](#)(int *query_identifieur*)

Identique à [mysql_num_fields\(\)](#).

[10.42.33 mysql_numrows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_numrows](#)

Identique à [mysql_num_rows\(\)](#).

10.42.34 mysql_pconnect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_pconnect](#)(string *hostname*)

Retourne un identifiant de connexion persistante à un serveur mSQL en cas de succès, et FALSE sinon.

[mysql_pconnect\(\)](#) se comporte presque comme [mysql_connect\(\)](#) mais avec deux différences majeures.

D'abord, lors de la connexion, cette fonction cherche si une connexion persistante a déjà été ouverte sur le même hôte. Si une telle connexion est trouvée, elle sera utilisée.

Deuxièmement, la connexion au serveur SQL ne sera pas terminée lors de la fin de l'exécution du script. A la place, le lien restera ouvert pour d'autres connexions futures. ([mysql_close\(\)](#) ne fermera pas un lien ouvert par [mysql_pconnect\(\)](#)).

C'est pourquoi une telle connexion est considérée comme 'persistante'.

10.42.35 mysql_query

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_query](#)(string *query*, int *link_identifier*)

[mysql_query\(\)](#) envoie une requête à la base de donnée active, sur le serveur associé à l'identifiant de connexion *link_identifier*. Si *link_identifier* n'est pas fourni, PHP tentera d'utiliser la dernière connexion ouverte. Si aucune connexion n'a été ouverte, la fonction tentera de se connecter par elle même, avec [mysql_connect\(\)](#) appelé sans argument.

Retourne un identifiant positif mSQL en cas de succès, et FALSE sinon.

Voir aussi: [mysql\(\)](#), [mysql_select_db\(\)](#), et [mysql_connect\(\)](#).

10.42.36 mysql_regcase

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Voir [sql_regcase\(\)](#).

10.42.37 mysql_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_result](#)(int *query_identifier*, int *i*, mixed *field*)

Retourne la valeur de la cellule, à la ligne *i* et l'offset spécifié, *field* dans le résultat mSQL *query_identifier*.

[mysql_result\(\)](#) retourne le contenu d'une cellule depuis un résultat mSQL *query_identifier*. L'argument de champs *field* peut être aussi bien un offset, qu'un nom de champs, ou encore le nom de la table point le nom du fichier (nom_table.nom_champs). Si la colonne est un alias, (par exemple 'select foo as bar from...'), utilisez de préférence l'alias au nom de colonne.

Lorsque vous travailler sur des résultats de grande taille, il est préférable d'utiliser les fonctions qui récupèrent toute la ligne (voir ci dessous). Comme ces fonctions retournent plusieurs cellules en même temps, elles sont beaucoup plus rapide que [mysql_result\(\)](#). De plus, sachez qu'accéder à un champs avec son indice numérique est beaucoup plus rapide qu'en utilisant les autres méthodes.

Alternatives recommandées : [mysql_fetch_row\(\)](#), [mysql_fetch_array\(\)](#) et [mysql_fetch_object\(\)](#).

[10.42.38 `mysql_select_db`](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_select_db](#)(string *database_name*, int *link_identifieur*)

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[mysql_select_db\(\)](#) choisit la base de données courante sur le serveur associé à l'identifiant de connexion *link_identifieur*. Si *link_identifieur* n'est pas fourni, PHP tentera d'utiliser la dernière connexion ouverte. Si aucune connexion n'a été ouverte, la fonction tentera de se connecter par elle-même, avec [mysql_connect\(\)](#) appelée sans argument.

Les prochains appels à [mysql_query\(\)](#) seront fait dans la base de données active.

Voir aussi: [mysql_connect\(\)](#), [mysql_pconnect\(\)](#) et [mysql_query\(\)](#).

[10.42.39 `mysql_selectdb`](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Voir [mysql_select_db\(\)](#).

[10.42.40 `mysql_tablename`](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_tablename](#)(int *query_identifieur*, int *field*)

[mysql_tablename\(\)](#) prend un pointeur de résultat (retourné par la fonction [mysql_list_tables\(\)](#)), ainsi qu'un index, et retourne le nom d'une table. La fonction [mysql_numrows\(\)](#) peut servir à déterminer le nombre de table dans le pointeur de résultat. *Exemple [mysql_tablename\(\)](#)*

```
<?php
mysql_connect ("localhost");
$result = mysql_list_tables ("wisconsin");
$i = 0;
while ($i < mysql_numrows ($result)) {
    $tb_names[$i] = mysql_tablename ($result, $i);
    echo $tb_names[$i] . "<BR>";
    $i++;
}
?>
```

[10.43 Fonctions Microsoft SQL Server](#)

[\[Notes en ligne\]](#)

[10.43.1 `mssql_close`](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_close](#)(int *link_identifieur*)

Retourne TRUE en cas de succès, ou FALSE sinon.

[mssql_close\(\)](#) ferme la connexion à la base MS SQL Server, qui était associé à l'identifiant *link_identifieur*. Si ce dernier n'est pas précisé, la dernière connexion ouverte sera fermée.

Notez qu'il n'est pas nécessaire de fermer les connexions non persistantes aux bases de données, car elles seront fermées automatiquement à la fin du script.

[mssql_close\(\)](#) ne peut pas fermer les liens persistants, générés par [mssql_pconnect\(\)](#).

Voir aussi : [mssql_connect\(\)](#) et [mssql_pconnect\(\)](#).

10.43.2 [mssql_connect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_connect](#)(string *servername* , string *username* , string *password*)

Retourne un identifiant positif de lien en cas de succès, et FALSE sinon.

[mssql_connect\(\)](#) établit une connexion à un serveur MS SQL server. Le nom du serveur *servername* doit être valide, comme défini dans les fichiers d'interface'.

Si un deuxième appel est fait à [mssql_connect\(\)](#) avec les mêmes arguments, un nouveau lien ne sera pas retourné, mais le lien déjà ouvert sera retourné.

Le lien avec le serveur sera fermé dès la fin du script, ce qui fait qu'on n'est pas obligé de fermer explicitement la connexion à la fin du script avec [mssql_close\(\)](#).

Voir aussi [mssql_pconnect\(\)](#), [mssql_close\(\)](#).

10.43.3 [mssql_data_seek](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_data_seek](#)(int *result_identifieur*, int *row_number*)

Retourne TRUE en cas de succès, FALSE en cas d'échec.

[mssql_data_seek\(\)](#) déplace le pointeur interne de ligne, dans le résultat *result_identifieur*, jusqu'à la ligne *row_number*. Le prochain appel à [mssql_fetch_row\(\)](#) retournera cette ligne.

Voir aussi : [mssql_data_seek\(\)](#).

10.43.4 [mssql_fetch_array](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_fetch_array](#)(int *result*)

Retourne un tableau qui contient les valeurs de la ligne lues, en cas de succès, et FALSE en cas d'échec.

[mssql_fetch_array\(\)](#) est une version améliorée de [mssql_fetch_row\(\)](#). En plus de stocker les données dans un tableau à index numérique, elle les stocke aussi dans un tableau associatif, en utilisant les noms de colonnes comme clé.

Une chose importante à noter est que [mssql_fetch_array\(\)](#) n'est PAS significativement plus lente que [mssql_fetch_row\(\)](#), tandis qu'elle apporte un confort appréciable.

Pour plus de détails, voyez [mssql_fetch_row\(\)](#).

10.43.5 mssql_fetch_field

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [mssql_fetch_field](#)(int *result*, int *field_offset*)

Retourne un objet contenant les informations sur un champs.

[mssql_fetch_field\(\)](#) sert à lire des informations spécifiques à un champs, dans un résultat de requête. Si l'offset du champs *field_offset* n'est pas précisé, le prochain champs sera analysé.

Les propriétés de l'objet sont :

- name – nom de la colonne. Si la colonne est le résultat d'une fonction, le nom de la colonne sera computed#N, où #N est un numéro de série.
- column_source – le nom de la table d'où la colonne est originaire.
- max_length – taille maximale de la colonne
- numeric – 1 si la colonne est numérique

Voir aussi [mssql_field_seek\(\)](#).

10.43.6 mssql_fetch_object

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_fetch_object](#)(int *result*)

Retourne un objet dont les propriétés contiennent les valeurs de la ligne, ou FALSE si il n'y a plus de ligne. [mssql_fetch_object\(\)](#) est similaire à [mssql_fetch_array\(\)](#), avec une différence : un objet est retourné, au lieu d'un tableau. Indirectement, cela signifie que vous ne pouvez accéder aux données que par leur nom de champs, et pas par leur offset (les nombres sont illégaux comme nom de propriété).

En terme de vitesse, cette fonction est identique à [mssql_fetch_array\(\)](#), quasiment aussi rapide que [mssql_fetch_row\(\)](#) (la différence est non significative).

Voir aussi : [mssql_fetch_array\(\)](#) et [mssql_fetch_row\(\)](#).

10.43.7 mssql_fetch_row

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mssql_fetch_row](#)(int *result*)

Retourne un tableau qui contient les valeurs de la ligne à lire, ou bien FALSE si il n'y a plus de lignes à lire. [mssql_fetch_row\(\)](#) lit une ligne dans le résultat *result* et place les valeurs dans un tableau. Chaque valeur est enregistré dans un élément du tableau, et les indices commencent à 0.

Les appels suivants à [mssql_fetch_row\(\)](#) retourneront la ligne suivante, ou bien FALSE s'il ne reste plus de lignes.

Voir aussi : [mssql_fetch_array\(\)](#), [mssql_fetch_object\(\)](#), [mssql_data_seek\(\)](#) et [mssql_result\(\)](#).

10.43.8 mssql_field_length

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_field_length](#)(int *result*, int *offset*)

[10.43.9 mssql_field_name](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_field_name](#) (int *result*, int *offset*)

[10.43.10 mssql_field_seek](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_field_seek](#) (int *result*, int *field_offset*)

Modifie la valeur du pointeur de champs. le prochain appel à [mssql_fetch_field\(\)](#) qui ne précisera pas de numéro de champs, le champs fixé par [mssql_field_seek\(\)](#) sera retournée.
Voir aussi : [mssql_fetch_field\(\)](#).

[10.43.11 mssql_field_type](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mssql_field_type](#) (int *result*, int *offset*)

[10.43.12 mssql_free_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_free_result](#) (int *result*)

[mssql_free_result\(\)](#) n'a besoin d'être appelé que si on craint d'utiliser trop de mémoire durant une opération. Toutes les ressources liées à un résultat seront libérés par [mssql_free_result\(\)](#).

[10.43.13 mssql_get_last_message](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mssql_get_last_message](#) (void)

[10.43.14 mssql_min_error_severity](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [mssql_min_error_severity](#) (int *severity*)

[10.43.15 mssql_min_message_severity](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [mssql_min_message_severity](#) (int *severity*)

[10.43.16 mssql_num_fields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_num_fields](#)(int *result*)

[mssql_num_fields\(\)](#) retourne le nombre de champs dans un résultat.

Voir aussi : [mssql_query\(\)](#), [mssql_fetch_field\(\)](#) et [mssql_num_rows\(\)](#).

[10.43.17 mssql_num_rows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_num_rows](#)(string *result*)

[mssql_num_rows\(\)](#) retourne le nombre de lignes dans un résultat.

Voir aussi : [mssql_query\(\)](#) et [mssql_fetch_row\(\)](#).

[10.43.18 mssql_pconnect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_pconnect](#)(string *servername* , string *username* , string)

Retourne un identifiant positif de lien MS SQL en cas de succès, et FALSE en cas d'erreur.

[mssql_pconnect\(\)](#) se comporte comme [mssql_connect\(\)](#) mais avec deux différences :

Premièrement, lors de la connexion, la fonction va commencer par rechercher un lien persistant déjà ouvert avec le même hôte, le même nom d'utilisateur, *username* et le même mot de passe *password*. Si un tel lien est trouvé, cet identifiant sera retourné, au lieu d'en ouvrir une autre connexion.

Deuxièmement, la connexion au serveur SQL ne sera pas fermée à la fin du script, mais restera ouverte, pour d'autres utilisations ultérieures ([mssql_close\(\)](#) ne fermera pas un lien établi avec [mssql_pconnect\(\)](#)).

C'est pourquoi ce type de lien est dit 'persistant'.

[10.43.19 mssql_query](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_query](#)(string *query*, int *link_identifier*)

Retourne un identifiant positif de résultat en cas de succès, ou FALSE sinon.

[mssql_query\(\)](#) envoie la requête au serveur courant, associé à l'identifiant *link_identifier* (ou la base par défaut, si il est omis). Si aucun lien n'est ouvert, [mssql_query\(\)](#) essaiera d'en ouvrir une, en appelant [mssql_connect\(\)](#).

Voir aussi : [mssql_select_db\(\)](#) et [mssql_connect\(\)](#).

[10.43.20 mssql_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_result](#)(int *result*, int *i*, mixed *field*)

Retourne la valeur de la colonne, à la ligne donnée, dans le résultat MS SQL, ou FALSE en cas d'erreur.

[mssql_result\(\)](#) retourne le contenu d'une des cellules d'un résultat MS SQL. Le nom du champs peut être son nom littéral ou son offset, ou encore, le nom de la table + "." + le nom du champs, ou encore la même chose avec le nom de la base de données. Si la colonne a été aliasée, utilisez le nom de l'alias plutôt que celui de la colonne.

Lorsque vous travaillez sur des résultats de grande taille, il vaut mieux utiliser les fonctions qui récupèrent toute une ligne (voir ci après). Comme ces fonctions lisent toutes les valeurs en une passe, elles sont EXTREMEMENT PLUS RAPIDES que [mssql_result\(\)](#). De plus, pensez que l'utilisation de l'offset numérique est beaucoup plus rapide que l'utilisation du nom de la colonne.

Alternatives recommandées : [mssql_fetch_row\(\)](#), [mssql_fetch_array\(\)](#) et [mssql_fetch_object\(\)](#).

[10.43.21 mssql_select_db](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_select_db](#)(string *database_name*, int *link_identifiant*)

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[mssql_select_db\(\)](#) sélectionne la base de données active. Si aucun identifiant de connexion n'est fourni, la fonction utilisera la dernière connexion ouverte. Si aucune connexion n'a été ouverte, la fonction essaiera d'en ouvrir une avec [mssql_connect\(\)](#), et de l'utiliser.

Tous les appels à [mssql_query\(\)](#) seront fait dans cette base.

Voir aussi : [mssql_connect\(\)](#), [mssql_pconnect\(\)](#) et [mssql_query\(\)](#).

[10.44 Fonctions MySQL](#)

[\[Notes en ligne\]](#)

Ces fonctions vous permettent d'accéder aux bases de données MySQL.

Plus d'informations sont disponibles à <http://www.mysql.com/>.

[10.44.1 mysql_affected_rows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_affected_rows](#)(int *link_identifiant*)

[mysql_affected_rows\(\)](#) retourne le nombre de lignes affectées lors de la dernière requête INSERT, UPDATE ou DELETE sur le serveur associé à l'identifiant de connexion. Si cet identifiant n'est pas précisé, cette fonction utilise la dernière connexion ouverte.

Si la dernière requête était un DELETE sans clause WHERE, tous les enregistrements ont été effacés, mais cette fonction va retourner 0.

Cette commande n'est pas possible après un SELECT, car elle ne fonctionne qu'après des commandes qui modifient les enregistrements. Pour connaître le nombre de ligne retournées par un SELECT, utilisez [mysql_num_rows\(\)](#).

[10.44.2 mysql_change_user](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_change_user](#)(string *user*, string *password*, string *database* , int *link_identifiant*)

[mysql_change_user\(\)](#) change l'utilisateur en cours de la session active, ou sur la connexion spécifiée avec l'option *link_identifier*. Si une base est spécifiée, elle deviendra la base par défaut de l'utilisateur. Si une erreur de connexion survient, la connexion en cours restera active.

Note : Cette fonction a été introduite dans PHP 3.0.13 et requiert MySQL 3.23.3 ou plus récent.

10.44.3 mysql_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_close](#) (int *link_identifier*)

Retourne TRUE en cas de succès, et FALSE sinon.

[mysql_close\(\)](#) ferme la connexion au serveur MySQL associée à l'identifiant *link_identifier*. Si cet identifiant n'est pas spécifié, cette commande s'applique à la dernière connexion ouverte.

Note : Notez que cette commande n'est pas nécessaire, car toutes les connexions non persistantes seront automatiquement fermées à la fin du script.

[mysql_close\(\)](#) ne ferme pas les connexions persistantes générées par [mysql_pconnect\(\)](#).

Exemple MySQL_close

```
<?php
$link = mysql_connect ("kraemer", "marliesle", "secret") {
or die ("Impossible de se connecter");
}
print ("Connexion réussie");
mysql_close ($link);
?>
```

Voir aussi [mysql_connect\(\)](#) et [mysql_pconnect\(\)](#).

10.44.4 mysql_connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_connect](#) (string *hostname :port :/path/to/socket* , string *username* , string *password*)

Retourne un identifiant positif de connexion en cas de succès, et sinon FALSE.

[mysql_connect\(\)](#) établit une connexion à un serveur MySQL. Tous les arguments sont optionnels, et s'ils manquent, les valeurs par défaut sont utilisées (('localhost', nom du propriétaire du process, mot de passe vide).

Le nom d'hôte peut aussi inclure un numéro de port, sous la forme : "hostname:port" ou un chemin jusqu'à une socket sous la forme ":/path/to/socket" pour l'hôte localhost. Note : *Le support des ":port" a été ajouté à partir de la version 3.0B4.*

Le support de ":/path/to/socket" a été ajouté à partir de la version 3.0.10.

Vous pouvez supprimer le message d'erreur de connexion en ajoutant un arobase " au nom de la fonction.

Si un second appel à [mysql_connect\(\)](#) est fait avec les mêmes arguments, PHP ne va pas ouvrir une nouvelle connexion, mais va retourner l'identifiant de la connexion déjà ouverte.

Le lien sera fermé automatiquement dès que l'exécution du script sera terminée, à moins d'être fermé explicitement avec [mysql_close\(\)](#).

Exemple MySQL connect

```
<?php
    $link = mysql_connect ("kraemer", "marliesle", "secret") {
or die ("Connexion impossible");
    }
print ("Connexion réussie");
mysql_close ($link);
?>
```

Voir aussi [mysql_pconnect\(\)](#) et [mysql_close\(\)](#).

10.44.5 mysql_create_db

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_create_db](#)(string *database name*, int *link_identifier*)

[mysql_create_db\(\)](#) tente de créer une nouvelle base de données sur le serveur associé à l'identifiant *link_identifier*, ou la dernière connexion ouverte.

Exemple de création de base MySQL

```
<?php
    $link = mysql_pconnect ("kron", "jutta", "geheim") {
or die ("Connexion impossible");
    }
if (mysql_create_db ("my_db")) {
print ("Base de données créée\n");
    } else {
printf ("Erreur lors de la création de la base: %s\n", mysql_error());
    }
?>
```

Pour des raisons de compatibilité ascendante, [mysql_createdb\(\)](#) est toujours utilisable.

Voir aussi [mysql_drop_db\(\)](#).

10.44.6 mysql_data_seek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_data_seek](#)(int *result_identifier*, int *row_number*)

Retourne TRUE en cas de succès, et FALSE sinon.

[mysql_data_seek\(\)](#) déplace le pointeur interne de résultat, dans le résultat associé à l'identifiant de résultat *result_identifier*. Il le fait pointer à la ligne *row_number*. Le prochain appel à [mysql_fetch_row\(\)](#) retournera cette ligne.

Row_number commence à 0.

Exemple MySQL data seek

```

<?php
    $link = mysql_pconnect ("kron", "jutta", "geheim") {
or die ("Connexion impossible");
    }
mysql_select_db ("samp_db") {
or die ("Selection de base impossible");
    }
    $query = "SELECT last_name, first_name FROM friends";
    $result = mysql_query ($query) {
or die ("Requête impossible");
    }
    # récupère les lignes dans l'ordre inverse
for ($i = mysql_num_rows ($result) - 1; $i >=0; $i--) {
if (!mysql_data_seek ($result, $i)) {
printf ("Impossible d'atteindre la ligne %d\n", $i);
continue;
    }
if(!($row = mysql_fetch_object ($result)))
continue;
printf ("%s %s<BR>\n", $row->last_name, $row->first_name);
    }
mysql_free_result ($result);
?>

```

[10.44.7 mysql_db_query](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_db_query](#) (string *database*, string *query*, int *link_identifiant*)

Retourne un identifiant de résultat si la requête réussit, et FALSE sinon.

[mysql_db_query\(\)](#) Sélectionne une base de données et exécute une requête. Si l'identifiant de lien *link_identifiant* n'est pas précisé, la fonction prendra par défaut la dernière base de données ouverte sur le serveur, et si elle n'en trouve pas, elle tentera de se connecter, en utilisant la fonction [mysql_connect\(\)](#), sans arguments.

Voir aussi [mysql_connect\(\)](#).

Pour des raisons de compatibilité ascendante, [mysql\(\)](#) peut aussi être utilisé.

[10.44.8 mysql_drop_db](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_drop_db](#) (string *database_name*, int *link_identifiant*)

Retourne TRUE en cas de succès, et FALSE sinon.

[mysql_drop_db\(\)](#) essaie d'effacer une base de données entière sur le serveur associé à l'identifiant de connexion *link_identifiant*.

Voir aussi [mysql_create_db\(\)](#). Pour des raisons de compatibilité ascendante, [mysql_drop_db\(\)](#) est toujours utilisable.

[10.44.9 mysql_errno](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_errno](#) (int *link_identifiant*)

Les erreurs qui sont remontées depuis le serveur MySQL ne sont plus des alertes. A la place, il faut utiliser cette fonction pour obtenir le numéro d'erreur.

```
<?php
mysql_connect("marliesle");
echo mysql_errno().": ".mysql_error()."<BR>";
mysql_select_db("nonexistentdb");
echo mysql_errno().": ".mysql_error()."<BR>";
$conn = mysql_query("SELECT * FROM nonexistenttable");
echo mysql_errno().": ".mysql_error()."<BR>";
?>
```

Voir aussi [mysql_error\(\)](#).

10.44.10 mysql_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_error](#)(int *link_identifieur*)

Les erreurs générées par MySQL ne se transforment plus en alerte. A la place, elles sont accessibles via ces fonctions :

```
<?php
mysql_connect("marliesle");
echo mysql_errno().": ".mysql_error()."<BR>";
mysql_select_db("nonexistentdb");
echo mysql_errno().": ".mysql_error()."<BR>";
$conn = mysql_query("SELECT * FROM nonexistenttable");
echo mysql_errno().": ".mysql_error()."<BR>";
?>
```

Voir aussi [mysql_errno\(\)](#).

10.44.11 mysql_fetch_array

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mysql_fetch_array](#)(int *result*, int *result_type*)

Retourne un tableau qui contient la ligne demandée, ou FALSE si il ne reste plus de ligne.

[mysql_fetch_array\(\)](#) est une version étendue de [mysql_fetch_row\(\)](#). En plus d'enregistrer les données sous forme d'un tableau à indice numérique, elle peut aussi les enregistrer dans un tableau associatif, en utilisant les noms des champs comme indices.

Si plusieurs colonnes ont le même nom, la dernière colonne aura la priorité. Pour accéder aux autres colonnes du même nom, vous devez utiliser l'index numériques, ou faire un alias pour chaque colonne.

```
select t1.f1 as foo t2.f1 as bar from t1, t2
```

Il est important de souligner que [mysql_fetch_array\(\)](#) n'est PAS plus lente que [mysql_fetch_row\(\)](#), tandis qu'elle ajoute un confort d'utilisation notable.

L'option *result_type* de [mysql_fetch_array\(\)](#) est une constante qui peut prendre les valeurs suivantes : MYSQL_ASSOC, MYSQL_NUM, et MYSQL_BOTH.

Pour plus de détails, voir aussi [mysql_fetch_row\(\)](#).

mysql_fetch_array

```
<?php
mysql_connect($host,$user,$password);
$result = mysql_db_query("database","select * from table");
while($row = mysql_fetch_array($result)) {
echo $row["user_id"];
echo $row["fullname"];
}
mysql_free_result($result);
?>
```

10.44.12 mysql_fetch_field

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [mysql_fetch_field](#) (int *result*, int *field_offset*)

Retourne un objet contenant les données.

[mysql_fetch_field\(\)](#) sert à obtenir des informations à propos des champs, dans certaines requêtes. Si l'offset du champs n'est pas spécifié, le champs suivant le dernier champs retourné, est retourné.

Les propriétés de l'objet sont :

- name – nom de la colonne
- table – nom de la table de la colonne
- max_length – taille maximale de la colonne
- not_null – 1 si la colonne ne peut pas être NULL (attribut NOT NULL)
- primary_key – 1 si la colonne est une clé primaire (attribut PRIMARY KEY)
- unique_key – 1 si la colonne est une clé unique (attribut UNIQUE)
- multiple_key – 1 si la colonne est une clé non-unique
- numeric – 1 si la colonne est numérique
- blob – 1 si la colonne est BLOB
- type – le type de la colonne
- unsigned – 1 si la colonne est non signé
- zerofill – 1 si la colonne est complétée par des zéros.

Voir aussi [mysql_field_seek\(\)](#)

10.44.13 mysql_fetch_lengths

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mysql_fetch_lengths](#) (int *result*)

Retourne un tableau avec la taille de chaque colonne de la dernière ligne retournée par [mysql_fetch_row\(\)](#), sinon FALSE.

[mysql_fetch_lengths\(\)](#) stocke les tailles de chaque colonne de la dernière ligne retournée par

[mysql_fetch_row\(\)](#), [mysql_fetch_array\(\)](#), et [mysql_fetch_object\(\)](#) dans un tableau, en commençant à la position.

Voir aussi [mysql_fetch_row\(\)](#).

10.44.14 mysql_fetch_object

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [mysql_fetch_object](#)(int *result*, int *result_typ*)

Retourne un objet dont les propriétés correspondent à une ligne d'un résultat, ou FALSE si il n'y a plus d'autres lignes.

[mysql_fetch_object\(\)](#) est identique à [mysql_fetch_array\(\)](#), à la différence qu'elle retourne un objet à la place d'un tableau. Vous pourrez ainsi accéder aux valeurs des champs par leur nom, mais plus par leur offset (les nombres ne sont pas des noms MySQL).

L'argument optionnel *result_typ* est une constante qui peut prendre les valeurs suivantes : MYSQL_ASSOC, MYSQL_NUM, et MYSQL_BOTH.

Concernant la vitesse, cette fonction est aussi rapide que [mysql_fetch_array\(\)](#), et presque aussi rapide que [mysql_fetch_row\(\)](#) (la différence est insignifiante) *mysql_fetch_object*

```
<?php
mysql_connect($host,$user,$password);
$result = mysql_db_query("database","select * from table");
while($row = mysql_fetch_object($result)) {
echo $row->user_id;
echo $row->fullname;
}
mysql_free_result($result);
?>
```

Voir aussi [mysql_fetch_array\(\)](#) et [mysql_fetch_row\(\)](#).

10.44.15 mysql_fetch_row

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mysql_fetch_row](#)(int *result*)

Retourne un tableau énuméré qui correspond à la ligne demandée, ou FALSE si il ne reste plus de ligne.

[mysql_fetch_row\(\)](#) va rechercher une ligne dans le résultat associé à l'identifiant de résultat spécifié. La ligne est retournée sous la forme d'un tableau. Chaque colonne est enregistré sous la forme d'un tableau commençant à la position 0.

Les appels suivants à [mysql_fetch_row\(\)](#) retourneront la ligne suivante dans le résultat, ou FALSE si il n'y a plus de ligne disponible.

Voir aussi [mysql_fetch_array\(\)](#), [mysql_fetch_object\(\)](#), [mysql_data_seek\(\)](#), [mysql_fetch_lengths\(\)](#), et [mysql_result\(\)](#).

10.44.16 mysql_field_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_field_name](#) (int *result*, int *field_index*)

[mysql_field_name\(\)](#) retourne le nom d'une colonne. Les arguments de la fonction sont un identifiant de résultat *result* et l'index du champs, ie. `mysql_field_name($result,2)`;
Retournera le nom du deuxième champs dans le résultat associé à \$result.
Pour des raisons de compatibilité ascendante, `mysql_fieldname()` peut encore être utilisé.

10.44.17 mysql_field_seek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_field_seek](#) (int *result*, int *field_offset*)

Place le pointeur de résultat sur le champs spécifié. Lors du prochain appel à [mysql_fetch_field\(\)](#) qui n'aura pas d'argument d'index de champs, le champs désormais pointé sera retourné.
Voir aussi [mysql_fetch_field\(\)](#).

10.44.18 mysql_field_table

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_field_table](#) (int *result*, int *field_offset*)

Retourne le nom de la table où se trouve une colonne. Pour des raisons de compatibilité ascendante, `mysql_fieldtable()` peut encore être utilisé.

10.44.19 mysql_field_type

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_field_type](#) (int *result*, int *field_offset*)

[mysql_field_type\(\)](#) est similaire à la fonction [mysql_field_name\(\)](#). Les arguments sont identiques, mais c'est le type du champs qui est retourné. Il vaudra "int", "real", "string", "blob", ou d'autres, comme détaillé dans la documentation MySQL. *Types mysql field*

```

<?php
mysql_connect("localhost:3306");
mysql_select_db("wisconsin");
$result = mysql_query("SELECT * FROM onek");
$fields = mysql_num_fields($result);
$rows = mysql_num_rows($result);
$i = 0;
$table = mysql_field_table($result, $i);
echo "Your '". $table. "' table has ". $fields. " fields et ". $rows. " records <BR>";
echo "The table has the following fields <BR>";
while ($i < $fields) {
    $type = mysql_field_type ($result, $i);

```

```

    $name = mysql_field_name ($result, $i);
    $len  = mysql_field_len  ($result, $i);
    $flags = mysql_field_flags ($result, $i);
echo $type." ".$name." ".$len." ".$flags."<BR>";
    $i++;
}
mysql_close();
?>

```

Pour des raisons de compatibilité ascendante, `mysql_fieldtype()` peut encore être utilisé.

[10.44.20 mysql field flags](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_field_flags](#)(int *result*, int *field_offset*)

[mysql_field_flags\(\)](#) retourne le sémaphore associé au champs spécifié par *field_offset*. Les sémaphores sont retournés comme des mots, séparés par des espaces, ce qui les rend facile à séparer, avec la commande [explode\(\)](#).

Les valeurs suivantes (pour une version suffisamment récente de MySQL) sont disponibles : "not_null", "primary_key", "unique_key", "multiple_key", "blob", "unsigned", "zerofill", "binary", "enum", "auto_increment", "timestamp".

Pour des raisons de compatibilité ascendante, `mysql_fieldflags()` peut encore être utilisé.

[10.44.21 mysql field len](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_field_len](#)(int *result*, int *field_offset*)

[mysql_field_len\(\)](#) retourne la taille du champs spécifié.

Pour des raisons de compatibilité ascendante, `mysql_fieldlen()` peut encore être utilisé.

[10.44.22 mysql free result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_free_result](#)(int *result*)

[mysql_free_result\(\)](#) n'est à appeler que si vous avez peur d'utiliser trop de mémoire durant l'exécution de votre script. Toute la mémoire associée à l'identifiant de résultat sera automatiquement libérée.

Pour des raisons de compatibilité ascendante, `mysql_freeresult()` peut encore être utilisé.

[10.44.23 mysql insert id](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_insert_id](#)(int *link_identifieur*)

[mysql_insert_id\(\)](#) retourne le dernier identifiant généré par un champs de type AUTO_INCREMENTED.

Cette fonction ne prend aucun argument. Elle retourne le dernier identifiant généré par la dernière fonction

INSERT effectuée.

[10.44.24 mysql_list_fields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_list_fields](#)(string *database_name*, string *table_name*, int *link_identifieur*)

[mysql_list_fields\(\)](#) recherche les informations à propos de la table spécifiée. Les arguments sont la base de données, et le nom de la table. Un pointeur de résultat est retourné, et pourra être passé à [mysql_field_flags\(\)](#), [mysql_field_len\(\)](#), [mysql_field_name\(\)](#) et [mysql_field_type\(\)](#).

Un identifiant de résultat est un entier positif. La fonction retourne -1 si une erreur survient. Une chaîne décrivant le problème rencontré sera placée dans la variable \$phperrormsg, et, à moins que la fonction n'ait été appelée sous la forme @mysql(), cette erreur sera aussi affichée.

Pour des raisons de compatibilité ascendante, mysql_listfields() est encore disponible.

[10.44.25 mysql_list_dbs](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_list_dbs](#)(int *link_identifieur*)

[mysql_list_dbs\(\)](#) retournera un identifiant de résultat, qui contiendra les noms des bases de données disponibles sur le serveur MySQL. Utilisez la fonction [mysql_tablename\(\)](#) pour lire toutes les bases de données.

Pour des raisons de compatibilité ascendante, mysql_listdbs() est encore disponible.

[10.44.26 mysql_list_tables](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_list_tables](#)(string *database*, int *link_identifieur*)

[mysql_list_tables\(\)](#) prend le nom d'une base de données comme argument, et retourne un identifiant de résultat, qui contiendra la liste des tables. La fonction [mysql_tablename\(\)](#) est le meilleur moyen d'extraire les noms des tables depuis l'identifiant de résultat.

Pour des raisons de compatibilité ascendante, mysql_listtables() est encore disponible.

[10.44.27 mysql_num_fields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_num_fields](#)(int *result*)

[mysql_num_fields\(\)](#) retourne le nombre de champs d'un résultat.

Voir aussi [mysql_db_query\(\)](#), [mysql_query\(\)](#), [mysql_fetch_field\(\)](#), [mysql_num_rows\(\)](#).

Pour des raisons de compatibilité ascendante mysql_numfields() est encore disponible.

[10.44.28 mysql_num_rows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_num_rows](#)(int *result*)

[mysql_num_rows\(\)](#) retourne le nombre de ligne d'une résultat.

Voir aussi [mysql_db_query\(\)](#), [mysql_query\(\)](#) et [mysql_fetch_row\(\)](#).

Pour des raisons de compatibilité ascendante [mysql_numrows\(\)](#) est encore disponible.

[10.44.29 mysql_pconnect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_pconnect](#)(string *hostname :port :/path/to/socket* , string *username* , string *password*)

Retourne un lien persistant positif en cas de succès, et sinon FALSE en cas d'erreur.

[mysql_pconnect\(\)](#) établit uen connexion persistante à un serveur MySQL. Tous les arguments sont optionnels, et des valeurs par défaut seront utilisés en cas d'omission ('localhost', nom d'utilisateur propriétaire du processus, mot de passe vide).

Le nom de l'hôte peut aussi inclure le numéro de port, c'est à dire "hostname:port" ou un chemin jusqu'à la socket ":/path/to/socket" pour l'hôte local. Note : *Le support de " :port" a été ajouté dans en version 3.0B4. Le support de " :/path/to/socket" a été ajouté dans en version 3.0.10.*

[mysql_pconnect\(\)](#) se comporte exactement comme [mysql_connect\(\)](#), mais avec deux différences majeures : Premièrement, lors de la connexion, la fonction essaie de trouver une connexion permanente déjà ouverte sur cet hote, avec le même nom d'utilisateur et de mot de passe. Si une telle connexion est trouvée, son identifiant est retourné, sans ouvrir de nouvelle connexion.

Deuxièmement, la connexion au serveur MySQL ne sera pas terminée avec la fin du script. Au lieu de cela, le lien sera conservé pour un prochain accès ([mysql_close\(\)](#) ne terminera pas une connexion persistante établie par [mysql_pconnect\(\)](#)).

C'est pourquoi ce type de connexion est dite 'persistante'.

[10.44.30 mysql_query](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_query](#)(string *query*, int *link_identifier*)

[mysql_query\(\)](#) envoie une requête SQL à la base de données actuellement active sur le serveur MySQL. Si *link_identifier* n'est pas précisé, la dernière connexion est utilisée. Si aucune connexion n'a été ouverte, la fonction tentera d'en ouvrir une, avec la fonction [mysql_connect\(\)](#) mais sans aucun paramètre (c'est à dire avec les valeurs par défaut).

[mysql_connect\(\)](#)

[mysql_query\(\)](#) retourne TRUE ou FALSE, pour indiquer le succès ou l'échec de la requête. En cas de retour TRUE, la requête était valide et a pu être exécuté sur le serveur. Cela n'indique pas le nombre de ligne affectées, ou retournées. Il est parfaitement possible qu'une requête valide n'affecte aucune ligne ou ne retourne aucune ligne.

L'exemple suivant est syntaxiquement invalide, ce qui conduit [mysql_query\(\)](#) à l'échec, et retourne FALSE:

[mysql_query\(\)](#)

```
<?php
$result = mysql_query ("SELECT * WHERE 1=1")
or die ("Invalid query");
?>
```

L'exemple suivant est sémantiquement invalide si `my_col` n'est pas une colonne de la table `my_tbl`, ce qui conduit [mysql_query\(\)](#) à l'échec, et retourne FALSE : [mysql_query\(\)](#).

```
<?php
$result = mysql_query ("SELECT my_col FROM my_tbl")
or die ("Invalid query");
?>
```

[mysql_query\(\)](#) échouera aussi et retournera aussi FALSE si les droits d'accès ne sont pas suffisants. En supposant que la requête réussisse, vous pouvez appeler [mysql_affected_rows\(\)](#) pour connaître le nombre de lignes affectées (pour les commandes DELETE, INSERT, REPLACE, ou UPDATE). Pour les commandes SELECT , [mysql_query\(\)](#) retourne un identifiant de résultat que vous pouvez passer à [mysql_result\(\)](#). Lorsque vous avez terminé avec le résultat, libérez la mémoire avec [mysql_free_result\(\)](#). Voir aussi [mysql_affected_rows\(\)](#), [mysql_db_query\(\)](#), [mysql_free_result\(\)](#), [mysql_result\(\)](#), [mysql_select_db\(\)](#) et [mysql_connect\(\)](#).

10.44.31 mysql_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_result](#) (int *result*, int *row*, mixed *field*)

[mysql_result\(\)](#) retourne le contenu d'un champs dans un résultat MySQL. L'argument de champs *row* peut être un offset de champs, ou le nom du champs, ou le nom de la table + point + le nom du champs (table.champs). Si la colonne a été aliassée, utilisez de préférence l'alias.

Lorsque vous travaillez sur des résultats de grande taille, vous devriez utiliser une des fonctions qui vont rechercher une ligne entière dans un tableau. Ces fonctions sont NETTEMENT plus rapides. De plus, l'utilisation d'un offset numériques est aussi beaucoup plus rapide que de spécifier un nom littéral. Les appels [mysql_result\(\)](#) ne devraient pas être mélangés avec d'autres fonctions qui travaillent aussi sur le résultat.

Alternatives à haut rendement, RECOMMANDÉES : [mysql_fetch_row\(\)](#), [mysql_fetch_array\(\)](#) et [mysql_fetch_object\(\)](#).

10.44.32 mysql_select_db

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_select_db](#) (string *database_name*, int *link_identif*)

Retourne TRUE en cas de succès, FALSE sinon.

[mysql_select_db\(\)](#) change la base de données active sur la connexion représentée par l'identifiant de connexion. Si aucun identifiant n'est spécifié, la dernière connexion est utilisée. S'il n'y a pas de dernière

connexion, la fonction tentera de se connecter seule, avec [mysql_connect\(\)](#) et les paramètres par défaut. Toutes les requêtes suivantes avec [mysql_query\(\)](#) seront faites avec la base de données active. Voir aussi [mysql_connect\(\)](#), [mysql_pconnect\(\)](#), et [mysql_query\(\)](#). Pour des raisons de compatibilité ascendante [mysql_selectdb\(\)](#) est encore disponible.

[10.44.33 mysql_tablename](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_tablename](#) (int *result*, int *i*)

[mysql_tablename\(\)](#) prend le pointeur de résultat obtenu avec [mysql_list_tables\(\)](#) ou bien un index entier, et retourne le nom de la table. La fonction [mysql_num_rows\(\)](#) peut être utilisée pour déterminer le nombre de tables dans le pointeur de résultat. *Exemple [mysql_tablename\(\)](#)*

```
<?php
mysql_connect ("localhost:3306");
$result = mysql_list_tables ("wisconsin");
$i = 0;
while ($i < mysql_num_rows ($result)) {
    $tb_names[$i] = mysql_tablename ($result, $i);
    echo $tb_names[$i] . "<BR>";
    $i++;
}
?>
```

[10.45 Réseau](#)

[\[Notes en ligne\]](#)

[10.45.1 checkdnsrr](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [checkdnsrr](#) (string *host*, string *type*)

Recherche l'enregistrement DNS de type *type* correspondant à l'hôte *host*. Retourne TRUE si un record a été trouvé, et FALSE en cas d'erreur ou d'échec.

type peut prendre les valeurs suivantes : A, MX, NS, SOA, PTR, CNAME, ou ANY. Par défaut, la valeur est : MX.

host peut être soit une adresse IP de la forme x.x.x.x avec x entre 0 et 256, soit un nom d'hôte.

Voir aussi [getmxrr\(\)](#), [gethostbyaddr\(\)](#), [gethostbyname\(\)](#), [gethostbyname\(\)](#) et la page de manuel [named\(8\)](#).

[10.45.2 closelog](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [closelog](#)

[closelog\(\)](#) ferme le pointeur qui sert à écrire dans l'historique système. L'utilisation de [closelog\(\)](#) est optionnelle.

[10.45.3 debugger_off](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [debugger_off](#)

Inactive le debugueur interne de PHP. Le debugueur est toujours en cours de développement.

[10.45.4 debugger_on](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [debugger_on](#)(string *address*)

Active le debugueur interne de PHP, et le connecte à l'adresse *address*. Le debugueur est toujours en cours de développement.

[10.45.5 define_syslog_variables](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void @xref{function.define-syslog-variables , , define_syslog_variables (void) }

Initialise toutes les constantes utilisées par les fonctions de syslog.
Voir aussi [openlog\(\)](#), [syslog\(\)](#) et [closelog\(\)](#).

[10.45.6 fsockopen](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fsockopen](#)(string *hostname*, int *port*, int *errno*, string *errstr*, double *timeout*)

Créer un flot de connexion à l'Internet (AF_INET) ou à un domaine Unix (AF_UNIX). Via Internet, cette fonction va ouvrir une socket de connexion TCP avec l'hôte *hostname* sur le port *port*. Via un domaine Unix, *hostname* représente le chemin jusqu'à la socket, et *port* doit être mis à 0. L'option *timeout* sert à donner une durée maximale à cet appel.

[fsockopen\(\)](#) retourne un pointeur de fichier qui peut être utilisé avec d'autres fonctions fichiers, telles que [fgets\(\)](#), [fgetss\(\)](#), [fputs\(\)](#), [fclose\(\)](#) et [feof\(\)](#).

Si l'appel échoue, [fsockopen\(\)](#) retourne FALSE, et si les options *errno* et *errstr* ont été fournies, elles contiennent désormais les raisons de l'échec. Si l'erreur retournée est 0 et que la fonction retourne FALSE, c'est une indication d'erreur. C'est probablement dû à une erreur d'initialisation de la socket. Notez que *errno* et *errstr* sont passées par référence.

Suivant les environnements, le type 'domaine Unix' ou l'option *timeout* ne sont pas toujours disponibles.

La socket sera ouverte par défaut en mode bloquant. Vous pouvez changer de mode en utilisant :

[set_socket_blocking\(\)](#). *fsockopen example*

```
$fp = fsockopen("www.php.net", 80, errno, 38,$errstr, 30);
```

```

if(!$fp) {
echo "$errstr ($errno)<br>\n";
} else {
fputs($fp, "GET / HTTP/1.0\n\n");
while(!feof($fp)) {
echo fgets($fp,128);
}
fclose($fp);
}

```

Voir aussi [pfsockopen\(\)](#)

[10.45.7 gethostbyaddr](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gethostbyaddr](#)(string *ip_address*)

Retourne le nom d'hôte correspondant à l'IP *ip_address*. Si une erreur survient, retourne *ip_address*.
Voir aussi [gethostbyname\(\)](#).

[10.45.8 gethostbyname](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gethostbyname](#)(string *hostname*)

Retourne l'adresse IP correspondant à l'hôte *hostname*.
Voir aussi [gethostbyaddr\(\)](#).

[10.45.9 gethostbyname1](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [gethostbyname1](#)(string *hostname*)

Retourne la liste d'IP correspondant à l'hôte *hostname*.
Voir aussi [gethostbyname\(\)](#), [gethostbyaddr\(\)](#), [checkdnsrr\(\)](#), [getmxrr\(\)](#) et la page 8 du manuel.

[10.45.10 getmxrr](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getmxrr](#)(string *hostname*, array *mxhosts*, array *weight*)

Effectue une recherche DNS pour obtenir les enregistrements MX de l'hôte *hostname*. Retourne TRUE si des enregistrements sont trouvés, et FALSE si une erreur est rencontrée, ou si la recherche échoue.

La liste des enregistrements MX est placée dans le tableau *mxhosts*. Si le tableau *weight* est fourni, il sera rempli par les informations de poids.

Voir aussi [checkdnsrr\(\)](#), [gethostbyname\(\)](#), [gethostbyname1\(\)](#), [gethostbyaddr\(\)](#), et la page 8 du manuel.

[10.45.11 getprotobyname](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getprotobyname](#)(string *name*)

[getprotobyname\(\)](#) retourne le numéro de protocole associé avec le nom de protocole *name*, comme dans ``/etc/protocols'`.

Voir aussi [getprotobynumber\(\)](#).

[10.45.12 getprotobynumber](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [getprotobynumber](#)(int *number*)

[getprotobynumber\(\)](#) retourne le numéro de protocole associé avec le nom de protocole *name*, comme dans ``/etc/protocols'`.

Voir aussi [getprotobyname\(\)](#).

[10.45.13 getservbyname](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getservbyname](#)(string *service*, string *protocol*)

[getservbyname\(\)](#) retourne le numéro de port associé à au service *service* et protocole *protocol*, comme dans ``/etc/services'`. *protocol* vaut soit tcp ou udp.

Voir aussi [getservbyport\(\)](#).

[10.45.14 getservbyport](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [getservbyport](#)(int *port*, string *protocol*)

[getservbyport\(\)](#) le service internet associé au port *port* pour le protocole *protocol* comme dans ``/etc/services'`. *protocol* vaut soit tcp ou udp.

Voir aussi [getservbyname\(\)](#).

[10.45.15 ip2long](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ip2long](#)(string *ip_address*)

[ip2long\(\)](#) génère une adresse IPv4 à partir de son équivalent numérique. *Exemple* [ip2long\(\)](#).

```
<?
$ip = gethostbyname("www.php.net");
```

```

$out = "Les URLs suivantes sont équivalentes :<br>\n";
$out .= "http://www.php.net/, http://".$ip."/, and http://".ip2long($ip)."/<br>\n";
echo $out;
?>

```

Voir aussi: [long2ip\(\)](#)

[10.45.16 long2ip](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [long2ip](#)(int *proper_address*)

[long2ip\(\)](#) génère une adresse IP (format aaa.bbb.ccc.ddd) à partir de sa représentation littérale.

Voir aussi: [ip2long\(\)](#)

[10.45.17 openlog](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [openlog](#)(string *ident*, int *option*, int *facility*)

[openlog\(\)](#) ouvre la connexion à l'historique système. La chaîne *ident* sera ajouté à chaque message. Les valeurs de *option* et *facility* sont données dans la section suivante. L'utilisation de [openlog\(\)](#) est optionnelle; cette fonction sera automatiquement appelée par [syslog\(\)](#) si nécessaire, et dans ce cas, l'identification sera mise par défaut à FALSE.

Voir aussi [syslog\(\)](#) et [closelog\(\)](#).

[10.45.18 pfsockopen](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pfsockopen](#)(string *hostname*, int *port*, int *errno*, string *errstr*, int *timeout*)

Cette fonction se comporte exactement comme [fsockopen\(\)](#) mais la connexion ouverte le reste, même après la fin du script. C'est la version persistante de [fsockopen\(\)](#).

[10.45.19 socket_get_status](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [socket_get_status](#)(resource *socket_get_status*)

[socket_get_status\(\)](#) retourne les informations sur la socket *socket_get_status*, et fournit la réponse sous la forme d'un tableau à quatre entrées:

- *timed_out* (bool) – La socket a expirée en attendant des données
- *blocked* (bool) – La socket a été bloquée
- *eof* (bool) – Indique un événement fin de fichier (EOF)
- *unread_bytes* (int) – Nombre d'octets restant dans les buffers de la socket.

Voir aussi [accept_connect\(\)](#), [bind\(\)](#), [connect\(\)](#), [listen\(\)](#), et [strerror\(\)](#).

10.45.20 set socket blocking

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [set_socket_blocking](#)(int *socket descriptor*, int *mode*)

Si *mode* est FALSE, la socket est mise en mode non bloquant, et si il est TRUE, la socket est mise en mode bloquant. Cela affecte des appels tels que [fgets\(\)](#) qui lisent depuis une socket. En mode non bloquant, un appel [fgets\(\)](#) retournera immédiatement toujours TRUE tandis qu'en mode bloquant, elle va attendre que des données arrivent pour répondre TRUE.

10.45.21 socket set timeout

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [socket_set_timeout](#)(int *socket descriptor*, int *seconds*, int *microseconds*)

[socket_set_timeout\(\)](#) fixe la durée de vie de la socket *socket descriptor*, exprimée comme la somme de *seconds* secondes et *microseconds* micro-secondes. *Exemple* [socket_set_timeout\(\)](#)

```
<?php
$fp = fsockopen("http://www.php.net", 80);
if(!$fp) {
    echo "Unable to open\n";
} else {
    fputs($fp, "GET / HTTP/1.0\n\n");
    $start = time();
    socket_set_timeout($fp, 2);
    $res = fread($fp, 2000);
    var_dump(socket_get_status($fp));
    fclose($fp);
    print $res;
}
?>
```

Cette fonction s'appelait @xref{function.set-socket-timeout , , set_socket_timeout()} mais elle est désormais obsolète.

Voir aussi: [fsockopen\(\)](#) et [fopen\(\)](#).

10.45.22 syslog

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [syslog](#)(int *priority*, string *message*)

[syslog\(\)](#) génère un message qui sera inscrit dans l'historique par le système. *priority* est une combinaison des valeurs d'accès et de niveau, qui seront décrites dans la prochaine section. Les derniers arguments sont le message à envoyer. Attention : les caractères %m seront remplacés par l'erreur (sous forme de chaîne), présente dans `errno`.

Pour plus d'informations sur l'historique, reportez vous au manuel Unix (syslog). Avec Windows NT, l'historique est pris en charge par Event Log.

10.46 Fonctions NIS

[\[Notes en ligne\]](#)

NIS (feu Yellow Pages / Pages jaunes) permet la gestion par le réseau de fichiers d'administration importants (tel un fichier de mot de passe). Pour plus d'informations, reportez vous au manuel NIS, ou à [Introduction to YP/NIS](#) Introduction to YP/NIS (en anglais). Il existe un livre en anglais [Managing NFS and NIS](#)" par Hal Stern.

Pour ajouter ces fonctionnalités, vous devez compiler PHP avec l'option `--with-yp`.

10.46.1 yp_get_default_domain

[\[Notes en ligne\]](#) [\[Exemples\]](#)

`int yp_get_default_domain(void)`

`yp_get_default_domain\(\)` retourne le nom de domaine NIS par défaut. Ce nom de domaine peut être utilisé pour les futurs appels NIS.

Un domaine NIS peut être décrit comme un regroupement de cartes NIS. Tous les hôtes qui ont besoin d'informations, s'attachent à un domaine. Référez vous aux documents cités en début de document pour plus de détails.

Exemple avec le domaine par défaut

```
<?php
    $domain = yp_get_default_domain();
    echo "Le domaine par défaut est : " . $domain;
?>
```

10.46.2 yp_order

[\[Notes en ligne\]](#) [\[Exemples\]](#)

`int yp_order(string domain, string map)`

`yp_order\(\)` retourne le numéro d'ordre d'une carte ou FALSE.

Exemple d'ordre NIS

```
<?php
    $number = yp_order($domain,$mapname);
    echo "Le numéro d'ordre de cette carte est : " . $order;
?>
```

Voir aussi [yp-get-default-domain\(\)](#).

[10.46.3 yp_master](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [yp_master](#)(string *domain*, string *map*)

[yp_master\(\)](#) retourne le nom de la machine maître d'une carte.

Exemple de maître NIS

```
<?php
    $number = yp_master($domain, $mapname);
echo "Master for this map is: " . $master;
?>
```

Voir aussi [yp_get_default_domain\(\)](#).

[10.46.4 yp_match](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [yp_match](#)(string *domain*, string *map*, string *key*)

[yp_match\(\)](#) retourne la valeur associée à la clé passée en argument, pour la carte spécifiée, ou FALSE. La clé doit exister et être exacte.

Exemple de recherche NIS

```
<?php
    $entry = yp_match($domain, "passwd.byname", "joe");
echo "La valeur trouvée est: " . $entry;
?>
```

Dans le cas présent, ce pourrait être: joe:##joe:11111:100:Joe User:/home/j/joe:/usr/local/bin/bash

Voir aussi [yp_get_default_domain\(\)](#).

[10.46.5 yp_first](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string[] [yp_first](#)(string *domain*, string *map*)

[yp_first\(\)](#) retourne le premier couple (clé ; valeur) d'une carte donnée, ou FALSE.

Exemple avec yp_first

```
<?php
    $entry = yp_first($domain, "passwd.byname");
    $key = key($entry);
```

```
echo "La première entrée de cette carte est " . $key
    . " et sa valeur est " . $entry[$key];
?>
```

Voir aussi [yp-get-default-domain\(\)](#).

10.46.6 yp_next

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string[] [yp_next](#)(string *domain*, string *map*, string *key*)

[yp_next\(\)](#) retourne le couple (clé ; valeur) suivant la clé donnée d'une carte donnée ou FALSE.

Exemple

```
<?php
    $entry = yp_next($domain, "passwd.byname", "joe");
    if(!$entry) {
    echo yp_errno() . ": " . yp_err_string();
    }
    $key = key($entry);
    echo "L'entree suivante après joe a la cle " . $key
        . " et sa valeur " . $entry[$key];
?>
```

Voir aussi [yp-get-default-domain\(\)](#).

10.47 Oracle 8 functions

[\[Notes en ligne\]](#)

Ces fonctions vous permettront d'accéder aux serveurs Oracle8 et Oracle7. Elles utilisent l'interface Oracle8 Call-Interface (OCI8). Vous aurez donc besoin des bibliothèques clientes Oracle8 pour pouvoir les utiliser. Il faut noter que cette extension est plus souple que l'extension Oracle officielle. Elle supporte notamment les liaisons entre les variables globales et locales de PHP avec des emplacements Oracle; elle supporte complètement les types LOB, FILE et ROWID et vous permet d'utiliser des variables de définitions personnalisables.

10.47.1 OCIDefineByName

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocidefinebyname](#)(int *stmt*, string *Column-Name*, mixed *&variable*, int *type*)

[ocidefinebyname\(\)](#) copie les valeurs issues de SQL-Columns dans des variables PHP. Méfiez-vous des colonnes Oracle qui sont toutes en majuscule, tandis que dans les SELECT, vous pouvez aussi les écrire en minuscule. [ocidefinebyname\(\)](#) s'attend à ce que *Column-Name* soit en majuscule. Si vous définissez une variable qui n'existe pas dans la commande SELECT, vous ne serez pas prévenu par une erreur. Si vous avez besoin de définir un type de données abstrait, tel que (LOB/ROWID/BFILE), vous devez lui allouer la mémoire avec [ocinewdescriptor\(\)](#). Reportez vous aussi à [ocibindbyname\(\)](#).

OCIDefineByName

```
<?php
/* Exemple OCIDefineByPos par thies@digicol.de (980219) */
$conn = OCILogon("scott","tiger");
$stmt = OCIParse($conn,"select empno, ename from emp");
/* La définition DOIT être faite AVANT ociexecute! */
OCIDefineByName($stmt,"EMPNO",empno);
OCIDefineByName($stmt,"ENAME",ename);
OCIExecute($stmt);
while (OCIFetch($stmt)) {
echo "empno:". $empno. "\n";
echo "ename:". $ename. "\n";
}
OCIFreeStatement($stmt);
OCILogoff($conn);
?>
```

10.47.2 OCIBindByName

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocibindbyname](#)(int *stmt*, string *ph_name*, mixed *&variable*, int *length*, int *type*)

[ocibindbyname\(\)](#) relie la variable PHP *variable* à l'emplacement Oracle *ph_name*. Son utilisation (comme entrée ou comme sortie) sera définie à l'exécution, et l'espace nécessaire sera alloué. Le paramètre de longueur *length* fixe la taille maximum pour la liaison. Si vous affectez une longueur de -1, [ocibindbyname\(\)](#) utilisera la longueur de variable comme maximum.

Si vous devez lier des types abstraits de données (LOB/ROWID/BFILE), vous devrez l'allouer dans un premier temps, avec [ocinewdescriptor\(\)](#). La longueur *length* ne sert pas pour ces types et devrait être fixée à -1. La variable *type* indique au serveur Oracle, quel type de pointeur va être utilisé. Les valeurs possibles sont : OCI_B_FILE (Fichier binaires), OCI_B_CFILE (Fichier texte), OCI_B_CLOB (LOB- texte), OCI_B_BLOB (LOB binaire) et OCI_B_ROWID (ROWID).

OCIDefineByName

```
<?php
/* Exemple OCIBindByPos par thies@digicol.de (980221)
insère 3 lignes dans emp, et utilise ROWID pour mettre à jour
les lignes, juste après l'insertion.
*/
$conn = OCILogon("scott","tiger");
$stmt = OCIParse($conn,"insert into emp (empno, ename) ".
                "values (:empno,:ename) ".
                "returning ROWID into :rid");
$data = array(1111 => "Larry", 2222 => "Bill", 3333 => "Jim");
$rowid = OCINewDescriptor($conn,OCI_D_ROWID);
OCIBindByName($stmt,":empno",empno,32);
OCIBindByName($stmt,":ename",ename,32);
OCIBindByName($stmt,":rid",rowid,-1,OCI_B_ROWID);
$update = OCIParse($conn,"update emp set sal = :sal where ROWID = :rid");
OCIBindByName($update,":rid",rowid,-1,OCI_B_ROWID);
OCIBindByName($update,":sal",sal,32);
$sal = 10000;
```

```

while (list($empno,$ename) = each($data)) {
OCIExecute($stmt);
OCIExecute($update);
}
$rowid->free();
OCIFreeStatement($update);
OCIFreeStatement($stmt);
$stmt = OCIParse($conn,"select * from emp where empno in (1111,2222,3333)");
OCIExecute($stmt);
while (OCIFetchInto($stmt,38;$arr,OCI_ASSOC)) {
var_dump($arr);
}
OCIFreeStatement($stmt);
/* delete our "junk" from the emp table... */
$stmt = OCIParse($conn,"delete from emp where empno in (1111,2222,3333)");
OCIExecute($stmt);
OCIFreeStatement($stmt);
OCILogout($conn);
?>

```

[10.47.3 OCILogon](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocilogon\(\)](#)(string *username*, string *password*, string *db*)

[ocilogon\(\)](#) retourne un identifiant de connexion, nécessaire à la plus part des fonctions OCI. Si l'option ORACLE_SID n'est pas précisée, PHP utilisera la variable d'environnement ORACLE_SID pour déterminer le serveur de connexion.

Les connexions sont partagées, à l'intérieur d'une même page avec [ocilogon\(\)](#). Cela signifie que COMMIT et ROLLBACK s'appliquent à toutes les transactions commencées à l'intérieur d'une même page, même si vous avez créé de multiples connexions.

Cet exemple montre comment les connexions sont partagées : *OCILogon*

```

<?php
print "<HTML><PRE>";
$db = "";
$c1 = ocilogon("scott","tiger",$db);
$c2 = ocilogon("scott","tiger",$db);
function create_table($conn)
{ $stmt = ociparse($conn,"create table scott.hallo (test
varchar2(64))");
ociexecute($stmt);
echo $conn." created table\n\n";
}
function drop_table($conn)
{ $stmt = ociparse($conn,"drop table scott.hallo");
ociexecute($stmt);
echo $conn." dropped table\n\n";
}
function insert_data($conn)
{ $stmt = ociparse($conn,"insert into scott.hallo values('$conn' || ' ' || to_char(sysdate,'DD-MO
ociexecute($stmt,OCI_DEFAULT);
echo $conn." inserted hallo\n\n";
}
function delete_data($conn)
{ $stmt = ociparse($conn,"delete from scott.hallo");

```

```

ociexecute($stmt,OCI_DEFAULT);
echo $conn." deleted hallo\n\n";
}
function commit($conn)
{ ocicommit($conn);
echo $conn." committed\n\n";
}
function rollback($conn)
{ ocirollback($conn);
echo $conn." rollback\n\n";
}
function select_data($conn)
{ $stmt = ociparse($conn,"select * from scott.hallo");
ociexecute($stmt,OCI_DEFAULT);
echo $conn."----selecting\n\n";
while (ocifetch($stmt))
echo $conn." <".ociresult($stmt,"TEST").">\n\n";
echo $conn."----done\n\n";
}
create_table($c1);
insert_data($c1); // Insertion d'une ligne avec c1
insert_data($c2); // Insertion d'une ligne avec c2
select_data($c1); // Les résultats des deux insertions sont retournés
select_data($c2);
rollback($c1); // Annulation avec c1
select_data($c1); // Les résultats des deux insertions sont annulés
select_data($c2);
insert_data($c2); // Insertion d'une ligne avec c2
commit($c2); // Validation avec using c2
select_data($c1); // Le résultat de c2 est retourné
delete_data($c1); // Effacement de toutes les lignes avec c1
select_data($c1); // Aucune ligne n'est retournée
select_data($c2); // Aucune ligne n'est retournée
commit($c1); // Validation avec c1
select_data($c1); // Aucune ligne n'est retournée
select_data($c2); // Aucune ligne n'est retournée
drop_table($c1);
print "</PRE></HTML>";
?>

```

Voir aussi [ociplogon\(\)](#) et [ocinlogon\(\)](#).

10.47.4 OCIPLogon

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ociplogon](#) (string *username*, string *password*, string *db*)

[ociplogon\(\)](#) crée une connexion persistante à un serveur Oracle 8 et s'authentifie. Si l'option ORACLE_SID n'est pas spécifiée, PHP utilisera la variable d'environnement ORACLE_SID pour déterminer le serveur de connexion.

Voir aussi [ocilogon\(\)](#) et [ocinlogon\(\)](#).

10.47.5 OCINLogon

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocinlogon](#)(string *username*, string *password*, string *db*)

[ocinlogon\(\)](#) crée une nouvelle connexion à un serveur Oracle et s'authentifie. Si l'option ORACLE_SID n'est pas spécifié, PHP utilisera la variable d'environnement ORACLE_SID pour déterminer le serveur de connexion.

[ocinlogon\(\)](#) force le serveur à établir une nouvelle connexion. Cette fonction ne doit être utilisée que si vous voulez isoler un ensemble de transactions. Par défaut, les connexions sont partagées au niveau de la page, si vous utilisez la fonction [ocinlogon\(\)](#) ou bien au niveau du processus web, si vous utilisez [ociplogon\(\)](#). Si vous avez de multiples connexions ouvertes avec [ocinlogon\(\)](#), les validations et annulations ne s'appliquent qu'à la connexion spécifiée.

L'exemple ci dessous montre l'utilisation des connexions séparées. *OCINLogon*

```
<?php
print "<HTML><PRE>";
$db = "";
$c1 = ocilogon("scott","tiger",$db);
$c2 = ocinlogon("scott","tiger",$db);
function create_table($conn)
{ $stmt = ociparse($conn,"create table scott.hallo (test
varchar2(64))");
ociexecute($stmt);
echo $conn." created table\n\n";
}
function drop_table($conn)
{ $stmt = ociparse($conn,"drop table scott.hallo");
ociexecute($stmt);
echo $conn." dropped table\n\n";
}
function insert_data($conn)
{ $stmt = ociparse($conn,"insert into scott.hallo values('$conn' || ' ' || to_char(sysdate,'DD-MO
ociexecute($stmt,OCI_DEFAULT);
echo $conn." inserted hallo\n\n";
}
function delete_data($conn)
{ $stmt = ociparse($conn,"delete from scott.hallo");
ociexecute($stmt,OCI_DEFAULT);
echo $conn." deleted hallo\n\n";
}
function commit($conn)
{ ocicommit($conn);
echo $conn." committed\n\n";
}
function rollback($conn)
{ ocirrollback($conn);
echo $conn." rollback\n\n";
}
function select_data($conn)
{ $stmt = ociparse($conn,"select * from scott.hallo");
ociexecute($stmt,OCI_DEFAULT);
echo $conn."----selecting\n\n";
while (ocifetch($stmt))
echo $conn." <".ociresult($stmt,"TEST").">\n\n";
echo $conn."----done\n\n";
}
create_table($c1);
insert_data($c1);
select_data($c1);
```

```

select_data($c2);
rollback($c1);
select_data($c1);
select_data($c2);
insert_data($c2);
commit($c2);
select_data($c1);
delete_data($c1);
select_data($c1);
select_data($c2);
commit($c1);
select_data($c1);
select_data($c2);
drop_table($c1);
print "</PRE></HTML>";
?>

```

Voir aussi [ocilogon\(\)](#) et [ociplogon\(\)](#).

10.47.6 OCILogOff

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocilogoff](#)(int *connection*)

[ocilogoff\(\)](#) ferme la connexion Oracle.

10.47.7 OCIExecute

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ociexecute](#)(int *statement*, int *mode*)

[ociexecute\(\)](#) exécute une commande déjà préparée (voir [ociparse\(\)](#)). L'option *mode* vous permet de spécifier le mode d'exécution (par défaut, il est à OCI_COMMIT_ON_SUCCESS). Si vous ne voulez pas que la commande soit automatiquement validée, utilisez le mode OCI_DEFAULT.

10.47.8 OCICommit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocicommit](#)(int *connection*)

[ocicommit\(\)](#) valide toutes les transactions en cours sur la connexion Oracle *connection*.

10.47.9 OCIRollback

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocirollback](#)(int *connection*)

[ocirollback\(\)](#) annule les transactions en cours sur la connexion Oracle *connection*.

10.47.10 OCINewDescriptor

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ocinewdescriptor](#) (int *connection*, int *type*)

[ocinewdescriptor\(\)](#) alloue l'espace nécessaire pour stocker un descripteur, ou un pointeur de LOB. Les valeurs acceptées pour type sont OCI_D_FILE, OCI_D_LOB et OCI_D_ROWID.

OCINewDescriptor

```

<?php
    /* Ce script est fait pour être appelé dans un formulaire HTML
    * Il attends les variables $user, $password, $table, $where, et $commitsize
    * Le scrip efface alors les lignes sélectionnées avec ROWID et valide
    * l'effacement après chaque groupe de $commitsize lignes.
    * (Utilisez avec prudence, car il n'y a pas d'annulation possible).
    */
    $conn = OCILogon($user, $password);
    $stmt = OCIParse($conn,"select rowid from $table $where");
    $rowid = OCINewDescriptor($conn,OCI_D_ROWID);
OCIDefineByName($stmt,"ROWID",rowid);
OCIExecute($stmt);
while ( OCIFetch($stmt) ) {
    $nrows = OCIRowCount($stmt);
    $delete = OCIParse($conn,"delete from $table where ROWID = :rid");
OCIBindByName($delete,":rid",rowid,-1,OCI_B_ROWID);
OCIExecute($delete);
print "$nrows\n";
if ( ($nrows % $commitsize) == 0 ) {
OCICommit($conn);
    }
    }
    $nrows = OCIRowCount($stmt);
print "$nrows effacées...\n";
OCIFreeStatement($stmt);
OCILogoff($conn);
?>

<?php
/* Ce script est fait pour être appelé depuis un formulaire HTML.
* Il attend les variables $user, $password, $table, $where, et $commitsize,
* données par le formulaire. Le script efface
* les lignes sélectionnées avec ROWID est valide les transactions
* à chaque jeu de $commitsize lignes. (Attention : il n'y plus d'annulation */
if(!isset($lob_upload) || $lob_upload == 'none'){
?>
<form action="upload.php3" method="post" enctype="multipart/form-data">
Upload file: <input type="file" name="lob_upload"><br>
<input type="submit" value="Upload"> - <input type="reset">
</form>
<?php
    } else {
        // $lob_upload contains the temporary filename of the uploaded file
        $conn = OCILogon($user, $password);
        $lob = OCINewDescriptor($conn, OCI_D_LOB);
        $stmt = OCIParse($conn,"insert into $table (id, the_blob) values(my_seq.NEXTVAL, EMPTY_BLOB(
OCIBindByName($stmt, ':the_blob', lob, -1, OCI_B_BLOB);
OCIExecute($stmt);

```

```

if($lob->savefile($lob_upload)){
OCICommit($conn);
echo "Blob sauvé!\n";
}else{
echo "Impossible de sauver le Blob\n";
}
OCIFreeDescriptor($lob);
OCIFreeStatement($stmt);
OCILogoff($conn);
}
?>

```

[10.47.11 OCIRowCount](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocirowcount](#)(int *statement*)

[ocirowcount\(\)](#) retourne le nombre de lignes affectées par une commande de modification. Cette fonction ne vous indiquera pas le nombre de lignes retournées par un SELECT : il faut que les lignes aient été modifiées.
OCIRowCount

```

<?php
print "<HTML><PRE>";
    $conn = OCILogon("scott","tiger");
    $stmt = OCIParse($conn,"create table emp2 as select * from emp");
OCIExecute($stmt);
print OCIRowCount($stmt) . " rows inserted.<BR>";
OCIFreeStatement($stmt);
    $stmt = OCIParse($conn,"delete from emp2");
OCIExecute($stmt);
print OCIRowCount($stmt) . " rows deleted.<BR>";
OCICommit($conn);
OCIFreeStatement($stmt);
    $stmt = OCIParse($conn,"drop table emp2");
OCIExecute($stmt);
OCIFreeStatement($stmt);
OCILogOff($conn);
print "</PRE></HTML>";
?>

```

[10.47.12 OCINumCols](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocinumcols](#)(int *stmt*)

[ocinumcols\(\)](#) retourne le nombre de colonnes dans un résultat
OCINumCols

```

<?php

```

```

print "<HTML><PRE>\n";
    $conn = OCILogon("scott", "tiger");
    $stmt = OCIParse($conn,"select * from emp");
OCIExecute($stmt);
while ( OCIFetch($stmt) ) {
print "\n";
    $ncols = OCINumCols($stmt);
for ( $i = 1; $i <= $ncols; $i++ ) {
    $column_name = OCIColumnName($stmt,$i);
    $column_value = OCIResult($stmt,$i);
print $column_name . ': ' . $column_value . "\n";
    }
print "\n";
}
OCIFreeStatement($stmt);
OCILogoff($conn);
print "</PRE>";
print "</HTML>\n";
?>

```

[10.47.13 OCIResult](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [ociresult](#)(int *statement*, mixed *column*)

[ociresult\(\)](#) retourne les données de la colonne *column* dans la ligne courante (voir [ocifetch\(\)](#)). [ocifetch\(\)](#) retournera tout les types, sauf les types abstraits (ROWIDs, LOBs et FILES).

[10.47.14 OCIFetch](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocifetch](#)(int *statement*)

[ocifetch\(\)](#) place la prochaine ligne (d'une commande SELECT) dans le pointeur interne de résultat.

[10.47.15 OCIFetchInto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocifetchinto](#)(int *stmt*, array *&result*, int *mode*)

[ocifetchinto\(\)](#) retourne la ligne suivante (pour une commande SELECT) dans le tableau *result*.

[ocifetchinto\(\)](#) crasera le contenu de *result*. Par défaut, *result* sera un tableau à index numérique, commençant à 1, et qui contiendra toute les colonnes qui ne sont pas NULL.

L'option *mode* vous permet de modifier le comportement par défaut de la fonction. Vous pouvez passer plusieurs modes simplement en les additionnant (ie OCI_ASSOC+OCI_RETURN_NULLS). Les modes valides sont :

- OCI_ASSOC Retourne un tableau associatif.
- OCI_NUM Retourne un tableau à index numérique (DEFAULT, valeur par défaut)
- OCI_RETURN_NULLS Retourne les colonnes vides.
- OCI_RETURN_LOBS Retourne la valeur des objets LOB plutôt que leur descripteur.

[10.47.16 OCIFetchStatement](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocifetchstatement\(\)](#) (int *stmt*, array &*variable*)

[ocifetchstatement\(\)](#) retourne toutes les lignes d'un résultat dans le tableau variable.

[ocifetchstatement\(\)](#) retourne le nombre de lignes retournées.

OCIFetchStatement

```

<?php
/* exemple OCIFetchStatement par mbritton@verinet.com (990624) */
$conn = OCILogon("scott","tiger");
$stmt = OCIParse($conn,"select * from emp");
OCIExecute($stmt);
$rows = OCIFetchStatement($stmt,$results);
if ( $rows > 0 ) {
print "<TABLE BORDER=\\"1\ ">\n";
print "<TR>\n";
while ( list( $key, $val ) = each( $results ) ) {
print "<TH>$key</TH>\n";
}
print "</TR>\n";
for ( $i = 0; $i < $rows; $i++ ) {
reset($results);
print "<TR>\n";
while ( $column = each($results) ) {
    $data = $column['value'];
print "<TD>$data[$i]</TD>\n";
}
print "</TR>\n";
}
print "</TABLE>\n";
} else {
echo "Rien n'a été trouvé<BR>\n";
}
print "$rows Records Selected<BR>\n";
OCIFreeStatement($stmt);
OCILogoff($conn);
?>

```

[10.47.17 OCIColumnsNULL](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocicolumnisnull\(\)](#) (int *stmt*, mixed *column*)

[ocicolumnisnull\(\)](#) retourne TRUE si la colonne *col* du résultat *stmt* est NULL. Vous pouvez utiliser le numéro de colonne (l'indexation des colonnes commence à 1) ou le nom de la colonne, pour le paramètre *col*.

10.47.18 OCIColumnSize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocicolumnsize](#) (int *stmt*, mixed *column*)

[ocicolumnsize\(\)](#) retourne la taille de la colonne. Vous pouvez utiliser l'index de colonne (l'indexation commence à 1) ou le nom de la colonne dans le paramètre *col*.

OCIColumnSize

```

<?php
print "<HTML><PRE>\n";
    $conn = OCILogon("scott", "tiger");
    $stmt = OCIParse($conn,"select * from emp");
OCIEExecute($stmt);
print "<TABLE BORDER=\\"1\">";
print "<TR>";
print "<TH>Name</TH>";
print "<TH>Type</TH>";
print "<TH>Length</TH>";
print "</TR>";
    $ncols = OCINumCols($stmt);
for ( $i = 1; $i <= $ncols; $i++ ) {
    $column_name = OCIColumnName($stmt,$i);
    $column_type = OCIColumnType($stmt,$i);
    $column_size = OCIColumnSize($stmt,$i);

print "<TR>";
print "<TD>$column_name</TD>";
print "<TD>$column_type</TD>";
print "<TD>$column_size</TD>";
print "</TR>";
}
print "</TABLE>";
OCIFreeStatement($stmt);
OCILogoff($conn);
print "</PRE>";
print "</HTML>\n";
?>

```

Voir aussi [ocinumcols\(\)](#), [ocicolumnname\(\)](#), et [ocicolumnsize\(\)](#).

10.47.19 OCI ServerVersion

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ociserverversion](#) (int *conn*)

[ociserverversion\(\)](#) retourne une chaîne contenant les informations de version du serveur **OCI ServerVersion**

```

<?php
    $conn = OCILogon("scott","tiger");
print "Version du serveur : " . OCIServerVersion($conn);

```

```
OCILogOff($conn);
?>
```

10.47.20 OCIStatementType

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ocistatementtype](#) (int *stmt*)

[ocistatementtype\(\)](#) retourne une des valeurs suivantes :

1. "SELECT"
2. "UPDATE"
3. "DELETE"
4. "INSERT"
5. "CREATE"
6. "DROP"
7. "ALTER"
8. "BEGIN"
9. "DECLARE"
10. "UNKNOWN"

Exemples

```
<?php
print "<HTML><PRE>";
    $conn = OCILogon("scott","tiger");
    $sql = "delete from emp where deptno = 10";
    $stmt = OCIParse($conn,$sql);
if ( OCIStatementType($stmt) == "DELETE" ) {
die "Vous n'etes pas autorisé à effacer dans cette table.<BR>";
}
OCILogoff($conn);
print "</PRE></HTML>";
?>
```

10.47.21 OCINewCursor

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocinewcursor](#) (int *conn*)

[ocinewcursor\(\)](#) alloue un nouveau pointeur de commande, pour la connexion *conn*.
Utiliser un REF CURSOR issue d'une procédure enregistrée.

```
<?php
// suppose your stored procedure info.output returns a ref cursor in :data
$conn = OCILogon("scott","tiger");
$curs = OCINewCursor($conn);
$stmt = OCIParse($conn,"begin info.output(:data); end;");
ocibindParam($stmt,"data",curs,-1,OCI_B_CURSOR);
ociexecute($stmt);
ociexecute($curs);
while (OCIFetchInto($curs,38;$data)) {
var_dump($data);
}
OCIFreeCursor($stmt);
OCIFreeStatement($curs);
OCILogoff($conn);
?>
```

Utiliser un REF CURSOR issue d'une commande SELECT

```
<?php
print "<HTML><BODY>";
$conn = OCILogon("scott","tiger");
$count_cursor = "CURSOR(select count(empno) num_emps from emp " .
                "where emp.deptno = dept.deptno) as EMPCNT from dept";
$stmt = OCIParse($conn,"select deptno,dname,$count_cursor");
ociexecute($stmt);
print "<TABLE BORDER=\\"1\">";
print "<TR>";
print "<TH>DEPT NAME</TH>";
print "<TH>DEPT #</TH>";
print "<TH># EMPLOYEES</TH>";
print "</TR>";
while (OCIFetchInto($stmt,38;$data,OCI_ASSOC)) {
print "<TR>";
    $dname = $data["DNAME"];
    $deptno = $data["DEPTNO"];
print "<TD>$dname</TD>";
print "<TD>$deptno</TD>";
ociexecute($data["EMPCNT"]);
while (OCIFetchInto($data["EMPCNT"],38;$subdata,OCI_ASSOC)) {
    $num_emps = $subdata["NUM_EMPS"];
print "<TD>$num_emps</TD>";
}
print "</TR>";
}
print "</TABLE>";
print "</BODY></HTML>";
OCIFreeStatement($stmt);
OCILogoff($conn);
?>
```

10.47.22 OCIFreeStatement

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocifreestatement](#) (int *stmt*)

[ocifreestatement\(\)](#) retourne TRUE en cas de succès, et FALSE en cas d'échec.

[10.47.23 OCIFreeCursor](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocifreecursor](#) (int *stmt*)

[ocifreecursor\(\)](#) retourne TRUE en cas de succès, et FALSE en cas d'échec.

[10.47.24 OCIColumnName](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ocicolumnname](#) (int *stmt*, int *col*)

[ocicolumnname\(\)](#) retourne le nom de la colonne correspondant au numéro de colonne passé en paramètre (l'indexation des colonnes commence à 1).

OCIColumnName

```
<?php
print "<HTML><PRE>\n";
    $conn = OCILogon("scott", "tiger");
    $stmt = OCIParse($conn,"select * from emp");
OCIExecute($stmt);
print "<TABLE BORDER=\\"1\">";
print "<TR>";
print "<TH>Name</TH>";
print "<TH>Type</TH>";
print "<TH>Length</TH>";
print "</TR>";
    $ncols = OCINumCols($stmt);
for ( $i = 1; $i <= $ncols; $i++ ) {
    $column_name = OCIColumnName($stmt,$i);
    $column_type = OCIColumnType($stmt,$i);
    $column_size = OCIColumnSize($stmt,$i);
print "<TR>";
print "<TD>$column_name</TD>";
print "<TD>$column_type</TD>";
print "<TD>$column_size</TD>";
print "</TR>";
    }
OCIFreeStatement($stmt);
OCILogoff($conn);
print "</PRE>";
print "</HTML>\n";
?>
```

Voir aussi [ocinumcols\(\)](#), [ocicolumntype\(\)](#), et [ocicolumnsize\(\)](#).

[10.47.25 OCIColumnType](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [ocicolumnname](#) (int *stmt*, int *col*)

[ocicolumntype\(\)](#) retourne le type de données de la colonne correspondant au numéro de colonne (les colonnes sont indexées à partir de 1).

OCIColumnType

```

<?php
print "<HTML><PRE>\n";
    $conn = OCILogon("scott", "tiger");
    $stmt = OCIParse($conn,"select * from emp");
OCIEExecute($stmt);
print "<TABLE BORDER=\\"1\">";
print "<TR>";
print "<TH>Name</TH>";
print "<TH>Type</TH>";
print "<TH>Length</TH>";
print "</TR>";
    $ncols = OCINumCols($stmt);
for ( $i = 1; $i <= $ncols; $i++ ) {
    $column_name = OCIColumnName($stmt,$i);
    $column_type = OCIColumnType($stmt,$i);
    $column_size = OCIColumnSize($stmt,$i);
print "<TR>";
print "<TD>$column_name</TD>";
print "<TD>$column_type</TD>";
print "<TD>$column_size</TD>";
print "</TR>";
}
OCIFreeStatement($stmt);
OCILogoff($conn);
print "</PRE>";
print "</HTML>\n";
?>

```

Voir aussi [ocinumcols\(\)](#), [ocicolumnname\(\)](#), et [ocicolumnsize\(\)](#).

[10.47.26 OCIParse](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ociparse](#) (int *conn*, string *query*)

[ociparse\(\)](#) analyse la requête *query* sur la connexion *conn*, et retourne TRUE si la requête *query* est valide, et FALSE, si ce n'est pas le cas. *query* peut être n'importe quelle requête SQL.

[10.47.27 OCIError](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ocierror](#) (int *stmt*)*conn*

[ocierror\(\)](#) retourne la dernière erreur trouvée. Si l'option *stmt/conn* n'est pas fournie, la dernière erreur rencontrée est retournée. Si aucune erreur n'est trouvée, [ocierror\(\)](#) retourne FALSE.

[10.47.28 OCIInternalDebug](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ociinternaldebug](#) (int *onoff*)

[ociinternaldebug\(\)](#) active l'affichage des informations de debuggage. Pour les afficher, mettez onoff à 0, ou sinon à 1 pour les cacher.

[10.48 Fonctions Oracle](#)

[\[Notes en ligne\]](#)

[10.48.1 Ora Bind](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_bind](#) (int *cursor*, string *PHP variable name*, string *SQL parameter name*, int *length*, int *type*)

[ora_bind\(\)](#) retourne TRUE si la liaison a pu se faire, et sinon FALSE. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

Cette fonction lie une variable PHP avec un paramètre SQL. Le paramètre SQL doit être de la forme ":name". Avec l'option, vous pouvez choisir si le paramètre SQL est de type entrée/sortie (0, valeur par défaut), entrée seulement (1) ou sortie seulement (2). Comme dans PHP 3.0.1, vous pouvez respectivement utiliser les constantes ORA_BIND_INOUT, ORA_BIND_IN et ORA_BIND_OUT plutôt que des nombres.

[ora_bind\(\)](#) doit être appelée après la fonction [ora_parse\(\)](#) et avant [ora_exec\(\)](#). Les valeurs d'entrées peuvent alors être fournies par assignation des variables PHP. Après la fonction [ora_exec\(\)](#) les variables liées contiennent les valeurs de sorties, si elles sont disponibles. Par exemple :

```
<?php
ora_parse($curs, "declare tmp INTEGER; begin tmp := :in; :out := tmp; :x := 7.77; end;");
ora_bind($curs, "result", ":x", $len, 2);
ora_bind($curs, "input", ":in", 5, 1);
ora_bind($curs, "output", ":out", 5, 2);
$input = 765;
ora_exec($curs);
echo "Résultat: $result<BR>Sortie: $output<BR>Entrée: $input";
?>
```

[10.48.2 Ora Close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_close](#) (int *cursor*)

Retourne TRUE si la fermeture a bien eu lieu, et FALSE sinon. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

Cette fonction termine les pointeurs ouverts avec la fonction [ora_open\(\)](#).

10.48.3 Ora_ColumnName

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ora_columnname\(\)](#) (int *cursor*, int *column*)

[ora_columnname\(\)](#) retourne le nom du champs *column* du pointeur *cursor*. Le nom retourné sera en majuscule.

10.48.4 Ora_ColumnType

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ora_columntype\(\)](#) (int *cursor*, int *column*)

[ora_columntype\(\)](#) retourne le type de la colonne *column* du résultat *cursor*. Le type retourné prendra une des valeurs suivantes :

- "VARCHAR2"
- "VARCHAR"
- "CHAR"
- "NUMBER"
- "LONG"
- "LONG RAW"
- "ROWID"
- "DATE"
- "CURSOR"

10.48.5 Ora_Commit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_commit\(\)](#) (int *conn*)

Retourne TRUE si la validation a bien eu lieu, et FALSE sinon. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

Cette fonction valide les transactions Oracle. Une transaction est définie par toutes les requêtes effectuées sur la connexion *conn* depuis la dernière validation ou annulation (avec auto-validation inactivée) ou depuis l'établissement de la connexion.

10.48.6 Ora_CommitOff

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_commitoff\(\)](#) (int *conn*)

[ora_commitoff\(\)](#) retourne TRUE si la désactivation a bien eu lieu, et FALSE sinon. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

Inactive la validation automatique après chaque [ora_exec\(\)](#).

10.48.7 Ora CommitOn

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_commiton](#)(int *conn*)

[ora_commiton\(\)](#) active la validation automatique après chaque [ora_exec\(\)](#).

Retourne TRUE si l'activation a bien eu lieu, et FALSE sinon. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

10.48.8 Ora Error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ora_error](#)(int *cursor_or_connection*)

Retourne un messages d'erreur de la forme *XXX-NNNNN* avec *XXX* qui est l'origine de l'erreur, et *NNNNN* qui identifie le message d'erreur. Note : *Le support des connexions a été ajouté dans PHP 3.0.4.*

Avec les versions UNIX d'Oracle, vous pouvez avoir des messages d'erreurs tels que :

```
$ oerr ora
00001 00001, 00000, "unique constraint (%s.%s) violated" // *Cause: An
update or insert statement attempted to insert a duplicate key // For
Trusted ORACLE configured in DBMS MAC mode, you may see // this message
if a duplicate entry exists at a different level. // *Action: Either
remove the unique restriction or do not insert the key .
```

10.48.9 Ora ErrorCode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_errorcode](#)(int *cursor_or_connection*)

[ora_errorcode\(\)](#) retourne le code d'erreur numérique de la dernière commande exécuté sur la connexion ou le pointeur fourni en paramètre. Note : *Les identifiants de connexion ne sont acceptés qu'à partir de la version 3.0.4.*

10.48.10 Ora Exec

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_exec](#)(int *cursor*)

[ora_exec\(\)](#) retourne TRUE en cas de succès, et FALSE en cas d'erreur. L'erreur générée sera alors accessible avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

10.48.11 Ora Fetch

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_fetch\(\)](#) (int *cursor*)

[ora_fetch\(\)](#) retourne TRUE (une ligne a été lue) ou FALSE (plus de lignes à lire ou erreur). Si une erreur survient, sa valeur sera disponible dans les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).
Lit une ligne de données sur le pointeur cursor.

10.48.12 Ora GetColumn

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [ora_getcolumn\(\)](#) (int *cursor*, mixed *column*)

[ora_getcolumn\(\)](#) retourne la valeur de la colonne. Si une erreur survient, FALSE est retourné et [ora_errorcode\(\)](#) aura une valeur non nulle. Notez, qu'un test à FALSE, avec cette fonction peut être TRUE, même sans erreur : en effet, la fonction peut retourner des valeurs telles que (résultat NULL, chaînes vides, nombre 0, la chaîne "0").

10.48.13 Ora Logoff

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_logoff\(\)](#) (int *connection*)

[ora_logoff\(\)](#) retourne TRUE si la fermeture a bien eu lieu, et FALSE sinon. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).
Déconnecte l'utilisateur, et se déconnecte.

10.48.14 Ora Logon

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_logon\(\)](#) (string *user*, string *password*)

[ora_logon\(\)](#) établit une connexion entre PHP et un serveur Oracle avec les noms d'utilisateur user et le mot de passe password.
Les connexions peut être faites avec *SQL*Net* en fournissant le nom *TNS* de la manière suivante :

```
$conn = Ora_Logon( "user@TNSNAME", "pass" );
```

Si vous avez des données qui ne sont pas ASCII, vous devriez vérifier que la variable *NLS_LANG* a été correctement configuré dans votre environnement. Pour les modules de serveur, vous devrez la configurer dans l'environnement d'exécution du serveur avant de le lancer.

Retourne un index de connexion, en cas de succès, ou FALSE en cas d'échec. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

10.48.15 Ora Open

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_open\(\)](#)(int *connection*)

[ora_open\(\)](#) ouvre un pointeur Oracle sur la connexion.

Retourne TRUE si la fermeture a bien eu lieu, et FALSE sinon. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

10.48.16 Ora Parse

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_parse\(\)](#)(int *cursor_ind*, string *sql_statement*, int *defer*)

[ora_parse\(\)](#) analyse une requête SQL ou un bloc PL/SQL et l'associe avec le pointeur *cursor_ind*. Retourne 0 en cas de succès, et -1 en cas d'erreur.

10.48.17 Ora Rollback

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_rollback\(\)](#)(int *connection*)

[ora_rollback\(\)](#) annule une transaction Oracle. (Voir [ora_commit\(\)](#) pour la définition d'une transaction). Retourne TRUE si la fermeture a bien eu lieu, et FALSE sinon. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

10.49 Output Control Functions

[\[Notes en ligne\]](#)

The Output Control functions allow you to control when output is sent from the script. This can be useful in several different situations, especially if you need to send headers to the browser after your script has began outputting data. The Output Control functions do not affect headers sent using [header\(\)](#) or [setcookie\(\)](#), only functions such as [echo\(\)](#) and data between blocks of PHP code.

Output Control example

```
<?php
ob_start();
echo "Hello\n";
setcookie ("cookiename", "cookiedata");
ob_end_flush();
?>
```

In the above example, the output from [echo\(\)](#) would be stored in the output buffer until [ob_end_flush\(\)](#) was called. In the mean time, the call to [setcookie\(\)](#) successfully stored a cookie without causing an error. (You can not normally send headers to the browser after data has already been sent.)

See also [header\(\)](#) and [setcookie\(\)](#).

[10.49.1 flush](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [flush](#)

Vide les buffers de sortie de PHP et tout ceux que PHP utilisait (CGI, un serveur web, etc.).

[10.49.2 ob_start](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ob_start](#)

This function will turn output buffering on. While output buffering is active no output is sent from the script, instead the output is stored in an internal buffer.

The contents of this internal buffer may be copied into a string variable using [ob_get_contents\(\)](#). To output what is stored in the internal buffer, use [ob_end_flush\(\)](#). Alternatively, [ob_end_clean\(\)](#) will silently discard the buffer contents.

See also [ob_get_contents\(\)](#), [ob_end_flush\(\)](#), [ob_end_clean\(\)](#), and [ob_implicit_flush\(\)](#)

[10.49.3 ob_get_contents](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ob_get_contents](#)

This will return the contents of the output buffer or FALSE, if output buffering isn't active.

See also [ob_start\(\)](#) and [ob_get_length\(\)](#).

[10.49.4 ob_get_length](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ob_get_length](#)

This will return the length of the contents in the output buffer or FALSE, if output buffering isn't active.

See also [ob_start\(\)](#) and [ob_get_contents\(\)](#).

[10.49.5 ob_end_flush](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ob_end_flush](#)

This function will send the contents of the output buffer (if any) and turn output buffering off. If you want to further process the buffer's contents you have to call [ob_get_contents\(\)](#) before [ob_end_flush\(\)](#) as the buffer contents are discarded after [ob_get_contents\(\)](#) is called.

See also [ob_start\(\)](#), [ob_get_contents\(\)](#), and [ob_end_clean\(\)](#).

10.49.6 ob_end_clean

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ob_end_clean](#)

This function discards the contents of the output buffer and turns off output buffering. See also [ob_start\(\)](#) and [ob_end_flush\(\)](#).

10.49.7 ob_implicit_flush

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ob_implicit_flush](#)(int *flag*)

[ob_implicit_flush\(\)](#) will turn implicit flushing on or off (if no *flag* is given, it defaults to on). Implicit flushing will result in a flush operation after every output call, so that explicit calls to [flush\(\)](#) will no longer be needed.

Turning implicit flushing on will disable output buffering, the output buffers current output will be sent as if [ob_end_flush\(\)](#) had been called.

See also [flush\(\)](#), [ob_start\(\)](#), and [ob_end_flush\(\)](#).

10.50 Ovrimos SQL

[\[Notes en ligne\]](#)

Ovrimos SQL Server est une base de données relationnelle client/serveur et transactionnelle, combinée avec des fonctionnalités web, et des transactions rapides.

Ovrimos SQL Server est disponible à www.ovrimos.com. Pour activer le support ovrimos de PHP, il suffit de compiler PHP avec l'option '--with-ovrimos' du script de configuration. Vous devrez aussi installer la librairie sqlcli disponible avec la distribution Ovrimos SQL Server.

Connection au serveur Ovrimos SQL Server et selectionn d'une table système ***Connection au serveur Ovrimos SQL Server et selectionn d'une table système***

```
<?php
$conn = ovrimos_connect ("server.domain.com", "8001", "admin", "password");
if ($conn != 0) {
echo ("Connection établie!");
    $res = ovrimos_exec ($conn, "select table_id, table_name from sys.tables");
if ($res != 0) {
echo "Requête effectuée!";
ovrimos_result_all ($res);
ovrimos_free_result ($res);
    }
ovrimos_close($conn);
}
?>
```

Cet exemple effectue une connexion réussie.

[10.50.1 ovrimos_connect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_connect](#)(string *host*, string *db*, string *user*, string *password*)

[ovrimos_connect\(\)](#) sert à se connecter à un serveur Ovrimos.

[ovrimos_connect\(\)](#) retourne un identifiant de connexion, supérieur à 0, ou 0 en cas d'échec. *host* est l'adresse IP de l'hôte Ovrimos, et *db* est soit le nom d'une base de données, soit une chaîne contenant le numéro de port.

Exemple avec [ovrimos_connect\(\)](#)

```

<?php
$conn = ovrimos_connect ("server.domain.com", "8001", "admin", "password");
if ($conn != 0) {
    echo "Connexion établie!";
    $res=ovrimos_exec ($conn, "select table_id, table_name from sys.tables");
    if ($res != 0) {
        echo "Requête effectuée!";
        ovrimos_result_all ($res);
        ovrimos_free_result ($res);
    }
    ovrimos_close ($conn);
}
?>

```

L'exemple ci dessus montre comment se connecter à une base de donnée et afficher le contenu d'une table.

[10.50.2 ovrimos_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ovrimos_close](#)(int *connection*)

[ovrimos_close\(\)](#) sert à fermer une connexion à un serveur Ovrimos.

[ovrimos_close\(\)](#) ferme la connexion au serveur Ovrimos. Toutes les transactions non validées sont annulées.

[10.50.3 ovrimos_close_all](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ovrimos_close_all](#)(void)

[ovrimos_close_all\(\)](#) sert à fermer toutes les connexions.

[ovrimos_close_all\(\)](#) ferme toutes les connexions à Ovrimos. Toutes les transactions non validées sont annulées.

[10.50.4 ovrimos_longreadlen](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_longreadlen](#)(int *result_id*, int *length*)

[ovrimos_longreadlen\(\)](#) sert à lire la taille des données qui sera lues lors de l'accès une colonne de grande taille.

[ovrimos_longreadlen\(\)](#) indique le nombre d'octets qui seront lus dans une colonne de grande taille (long varchar et long varbinary). Par défaut, 0. Indépendamment du fait que cette fonction requiert *result_id*, actuellement cette fonction affecte ce paramètre pour tous les résultats. Retourne vrai.

10.50.5 ovrivos_prepare

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_prepare](#)(int *connection_id*, string *query*)

[ovrimos_prepare\(\)](#) sert à préparer une requête SQL.

[ovrimos_prepare\(\)](#) prépare une requête SQL et retourne un identifiant de résultat *result_id* (ou FALSE en cas d'échec).

Connexion à un serveur Ovrivos SQL Server et préparation d'une requête

```
<?php
$conn=ovrimos_connect ("db_host", "8001", "admin", "password");
if ($conn!=0) {
echo "Connexion établie!";
    $res=ovrimos_prepare ($conn, "select table_id, table_name
from sys.tables where table_id=1");
if ($res != 0) {
echo "Préparation faite!";
if (ovrimos_execute ($res)) {
echo "Exécution réussie!\n";
ovrimos_result_all ($res);
    } else {
echo "Exécution manquée!";
    }
ovrimos_free_result ($res);
    } else {
echo "Préparation manquée!\n";
    }
ovrimos_close ($conn);
}
?>
```

Cet exemple montre comment se connecter à un serveur Ovrivos SQL Server, comment préparer une requête SQL et l'exécuter.

10.50.6 ovrivos_execute

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_execute](#)(int *result_id*, array *parameters_array*)

[ovrimos_execute\(\)](#) sert à exécuter une requête SQL.

[ovrimos_execute\(\)](#) exécute une requête préparée. Retourne true ou false. Si la requête préparée contient des paramètres (des points d'interrogations dans la requête), un nombre correct de paramètre doit être passé dans le tableau *parameters_array*. Notez que cette fonction ne suit pas les conventions PHP qui placent les noms des paramètres entre crochets. L'auteur n'a pas pu s'y faire.

10.50.7 ovrimos_cursor

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_cursor\(\)](#) (int *result_id*)

[ovrimos_cursor\(\)](#) sert à lire le nom du curseur

[ovrimos_cursor\(\)](#) retourne le nom du curseur. Pratique, lorsqu'on veut faire des modifications ou des effacements avec des curseurs déjà positionnés.

10.50.8 ovrimos_exec

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_exec\(\)](#) (int *connection_id*, string *query*)

[ovrimos_exec\(\)](#) sert à exécuter une requête SQL.

[ovrimos_exec\(\)](#) exécute une requête SQL (selection ou modification), et retourne un identifiant de résultat *result_id* (ou bien false, en cas d'échec). Evidemment, la requête SQL ne doit pas contenir de paramètres.

10.50.9 ovrimos_fetch_into

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_fetch_into\(\)](#) (int *result_id*, array *result_array*, string *how* , int *rownumber*)

[ovrimos_fetch_into\(\)](#) lit une ligne dans un résultat SQL.

[ovrimos_fetch_into\(\)](#) lit une ligne dans le résultat *result_id*, qui doit être passé en référence. La ligne qui sera lue est déterminée par les deux paramètres *how* et *rownumber*. *how* peut prendre les valeurs de 'Next' (suivant, valeur par défaut), 'Prev' (précédent), 'First' (premier), 'Last' (dernier), 'Absolute' (position absolue). La casse de *how* n'est pas prise en compte. *rownumber* est optionne, sauf dans le cas d'Absolute'. Retourne TRUE ou FALSE.

Lit un exemple

```

<?php
$conn=ovrimos_connect ("neptune", "8001", "admin", "password");
if ($conn!=0) {
echo "Connection établie!";
    $res=ovrimos_exec ($conn,"select table_id, table_name from sys.tables");
if ($res != 0) {
echo "Requête effectuée!";
if (ovrimos_fetch_into ($res, 38;$row)) {
list ($table_id, $table_name) = $row;
echo "table_id=".$table_id.", table_name=".$table_name."\n";
if (ovrimos_fetch_into ($res, 38;$row)) {
list ($table_id, $table_name) = $row;
echo "table_id=".$table_id.", table_name=".$table_name."\n";
        } else {
        }
echo "Next: erreur\n";
        }
    } else {

```

```

echo "First: erreur\n";
    }
ovrimos_free_result ($res);
    }
ovrimos_close ($conn);
}
?>

```

Cet exemple lis une ligne.

[10.50.10 ovrimos_fetch_row](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_fetch_row](#) (int *result_id*, int *how* , int *row_number*)

[ovrimos_fetch_row\(\)](#) lit une ligne dans un résultat SQL.

[ovrimos_fetch_row\(\)](#) lit une ligne dans un résultat. Les colonnes doivent être lues par un autre appel.

Retourne true ou false.

Exemple de lecture de ligne

```

<?php
$conn = ovrimos_connect ("remote.host", "8001", "admin", "password");
if ($conn != 0) {
echo "Connection établie!";
    $res=ovrimos_exec ($conn, "select table_id, table_name from sys.tables");
if ($res != 0) {
echo "Requête effectuée!";
if (ovrimos_fetch_row ($res, "First")) {
    $table_id = ovrimos_result ($res, 1);
    $table_name = ovrimos_result ($res, 2);
echo "table_id=".$table_id.", table_name=".$table_name."\n";
if (ovrimos_fetch_row ($res, "Next")) {
    $table_id = ovrimos_result ($res, "table_id");
    $table_name = ovrimos_result ($res, "table_name");
echo "table_id=".$table_id.", table_name=".$table_name."\n";
    } else {
echo "Next: erreur\n";
    }
    } else {
echo "First: erreur\n";
    }
ovrimos_free_result ($res);
    }
ovrimos_close ($conn);
}
?>

```

Cet exemple lit une ligne et l'affiche.

[10.50.11 ovrimos_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_result](#) (int *result_id*, mixed *field*)

[ovrimos_result\(\)](#) sert à lire le contenu d'une colonne.

[ovrimos_result\(\)](#) lit le contenu de la colonne *field* dans le résultat *result_id*. *field* peut être le nom de la colonne (une chaîne) ou bien le numéro de la colonne (la première colonne est alors 1).

10.50.12 ovrivos_result_all

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_result_all](#)(int *result_id*, string *format*)

[ovrimos_result_all\(\)](#) sert à afficher tout le résultat d'une requête.

[ovrimos_result_all\(\)](#) affiche le résultat de la requête représentée par *result_id*. Retourne TRUE ou FALSE.

Prépare une requête, l'exécute, et affiche le résultat

```
<?php
$conn = ovrivos_connect ("db_host", "8001", "admin", "password");
if ($conn != 0) {
echo "Connection établie!";
    $res = ovrivos_prepare ($conn, "select table_id, table_name
from sys.tables where table_id = 7");
if ($res != 0) {
echo "Préparation faite!";
if (ovrivos_execute ($res, array(3))) {
echo "Exécution réussie!\n";
ovrivos_result_all ($res);
    } else {
echo "Exécution manquée!";
    }
ovrivos_free_result ($res);
    } else {
echo "Préparation manquée!\n";
    }
ovrivos_close ($conn);
}
?>
```

Cet exemple exécute une requête SQL et affiche le résultat sous forme d'une table HTML.

[ovrimos_result_all\(\)](#) avec meta-information

```
<?php
$conn = ovrivos_connect ("db_host", "8001", "admin", "password");
if ($conn != 0) {
echo "Connection établie!";
    $res = ovrivos_exec ($conn, "select table_id, table_name
from sys.tables where table_id = 1");
if ($res != 0) {
echo "Requête effectuée! cursor=".ovrivos_cursor ($res)."\n";
    $colnb = ovrivos_num_fields ($res);
echo "Output columns=".$colnb."\n";
for ($i=1; $i<=$colnb; $i++) {
    $name = ovrivos_field_name ($res, $i);
    $type = ovrivos_field_type ($res, $i);
    $len = ovrivos_field_len ($res, $i);
echo "Colonne ".$i." nom=".$name." type=".$type." longueur=".$len."\n";
}
}
}
?>
```

```

    }
    ovrimos_result_all ($res);
    ovrimos_free_result ($res);
    }
    ovrimos_close ($conn);
}
?>

```

Exemple avec [ovrimos_result_all\(\)](#)

```

<?php
$conn = ovrimos_connect ("db_host", "8001", "admin", "password");
if ($conn != 0) {
echo "Connection établie!";
    $res = ovrimos_exec ($conn, "update test set i=5");
if ($res != 0) {
echo "Requête effectuée!";
echo ovrimos_num_rows ($res). " lignes affectées\n";
ovrimos_free_result ($res);
    }
ovrimos_close ($conn);
}
?>

```

10.50.13 ovrimos_num_rows

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_num_rows](#) (int *result_id*)

[ovrimos_num_rows\(\)](#) retourne le nombre de lignes affectées par une modification

10.50.14 ovrimos_num_fields

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_num_fields](#) (int *result_id*)

[ovrimos_num_fields\(\)](#) indique le nombre de colonnes du résultat *result_id*.

10.50.15 ovrimos_field_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_field_name](#) (int *result_id*, int *field_number*)

[ovrimos_field_name\(\)](#) sert à obtenir le nom d'une colonne.

[ovrimos_field_name\(\)](#) retourne le nom d'une colonne à partir de son numéro de colonne *field_number*, (la première colonne est à 1).

[10.50.16 ovrimos_field_type](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_field_type](#)(int *result_id*, int *field_number*)

[ovrimos_field_type\(\)](#) sert à connaître le type numérique d'une colonne.

[ovrimos_field_type\(\)](#) retourne le type numérique d'une colonne, identifiée par son numéro *field_number* dans le résultat *field_number*.

[10.50.17 ovrimos_field_len](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_field_len](#)(int *result_id*, int *field_number*)

[ovrimos_field_len\(\)](#) sert à connaître la taille d'une colonne.

[ovrimos_field_len\(\)](#) retourne la taille de la colonne *field_number*, dans le résultat *field_number*.

[10.50.18 ovrimos_field_num](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_field_num](#)(int *result_id*, string *field_name*)

[ovrimos_field_num\(\)](#) sert à connaître le numéro de colonne, à partir de son nom.

[ovrimos_field_num\(\)](#) retourne le numéro de la colonne *field_name* (la numérotation commence à 1), dans *result_id*.

[10.50.19 ovrimos_free_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_free_result](#)(int *result_id*)

[ovrimos_free_result\(\)](#) sert à effacer un résultat.

[ovrimos_free_result\(\)](#) libère toutes les ressources prises par le résultat *result_id*. Retourne TRUE.

[10.50.20 ovrimos_commit](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_commit](#)(int *connection_id*)

[ovrimos_commit\(\)](#) sert à exécuter une transaction.

[ovrimos_commit\(\)](#) exécute la transaction préparée sur la connexion *connection_id*.

[10.50.21 ovrimos_rollback](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_rollback](#)(int *connection_id*)

[ovrimos_rollback\(\)](#) sert à annuler une transaction.

[ovrimos_rollback\(\)](#) annule la transaction préparée sur la connexion *connection_id*.

10.51 Expressions régulières compatibles Perl

[\[Notes en ligne\]](#)

La syntaxe des masques utilisés dans ces fonctions ressemble fort à celle de Perl. Les expressions seront entourées de délimiteurs, slash (/), par exemple. N'importe quel caractère peut servir de délimiteur, tant qu'il n'est pas alphanumérique ou n'est pas un backslash (\). Si un délimiteur doit être utilisé dans l'expression, il faudra l'échapper avec un backslash.

Le délimiteur final peut être suivi d'options qui affecteront la recherche. [10.51.7 Options de recherche](#).

Exemples de masques valides

- `/<\w+>/`
- `|(\d{3})-\d+|Sm`
- `/^(?i)php[34]/`

Exemples de masques invalides

- `/href='(.*)'` – délimiteur final manquant
- `\w+\s*\w+/J` – option 'J' inconnue
- `1-\d3-\d3-\d4|` – délimiteur initial manquant

Note : *Les expressions régulières Perl sont disponibles depuis la PHP 4 et PHP 3.0.9.*

10.51.1 preg_match

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [preg_match](#)(string *pattern*, string *subject*, array *matches*)

[preg_match\(\)](#) analyse *subject* pour trouver l'expression *pattern*.

Si *matches* est fourni, il sera rempli par les résultats de la recherche. `$matches[0]` contiendra le texte qui satisfait le masque complet, `$matches[1]` contiendra le texte qui satisfait la première parenthèse capturante, etc..

Retourne TRUE si la recherche à réussie, et FALSE sinon (notamment en cas d'erreur).

Extraction d'un numéro de page d'une chaîne.

```
if (preg_match("/page\s+(\d+)/i", "Go to page #9.", $parts))
print "La page suivante est $parts[1]";
else
print "Page introuvable.";
```

Voir aussi [preg_match_all\(\)](#), [preg_replace\(\)](#) et [preg_split\(\)](#).

10.51.2 preg_match_all

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [preg_match_all](#)(string *pattern*, string *subject*, array *matches*, int *order*)

[preg_match_all\(\)](#) analyse *subject* pour trouver l'expression *pattern* et met les résultats dans *matches*, dans l'ordre spécifié par *order*.

Après avoir trouvé un premier résultat, la recherche continue jusqu'à la fin de la chaîne.

order peut prendre une des deux valeurs suivantes :

PREG_PATTERN_ORDER

- L'ordre est tel que \$matches[0] est un tableau qui contient les résultats qui satisfont le masque complet, \$matches[1] est un tableau qui contient les résultats qui satisfont la première parenthèse capturante, etc..

```
preg_match_all("|<[^>]+>(.*?)</[^>]+>|U", "<b>exemple: </b><div align=left>a test</div>", $
print $out[0][0].", ".$out[0][1]."\n";
print $out[1][0].", ".$out[1][1]."\n"
```

Cet exemple va afficher :

```
<b>exemple: <b38;gt; , <div align=left>ceci est un test</div>
exemple: , this is a test
```

Ainsi, \$out[0] est un tableau qui contient les résultats qui satisfont le masque complet, et \$out[1] est un tableau qui contient les balises entre < et >.

PREG_SET_ORDER

- Les résultats sont classés de telle façon que \$matches[0] contient la première série de résultat, \$matches[1] contient la deuxième série de résultat, etc..

```
preg_match_all("|<[^>]+>(.*?)</[^>]+>|U", "<b>exemple: </b><div align=left>a test</div>", $
print $out[0][0].", ".$out[0][1]."\n";
print $out[1][0].", ".$out[1][1]."\n"
```

Cet exemple va afficher :

```
<b>exemple: </b>, exemple:
<div align=left>this is a test</div>, this is a test
```

Dans ce cas, \$matches[0] est la première série de résultat, et \$matches[0][0] contient le texte qui satisfait le masque complet, \$matches[0][1] contient le texte de la première parenthèse capturante, etc. De même, \$matches[1] contient le texte qui satisfait le masque complet, etc. *

Si *order* est omis, PREG_PATTERN_ORDER est utilisé par défaut.

Retourne le nombre de résultat qui satisfont le masque complet, ou FALSE en cas d'échec ou d'erreur.

Extraction de tous les numéros de téléphone d'un texte.

```
preg_match_all("/\((? (\d{3})? \)? (?1) [\-\s] ) \d{3}-\d{4}/x",
    "Call 555-1212 or 1-800-555-1212", $phones);
```

Voir aussi [preg_match\(\)](#), [preg_replace\(\)](#) et [preg_split\(\)](#).

10.51.3 [preg_replace](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [preg_replace](#) (mixed *pattern*, mixed *replacement*, mixed *subject*)

[preg_replace\(\)](#) analyse *subject* pour trouver l'expression *pattern* et remplace les résultats par *replacement*. *replacement* peut contenir des références de la forme `\n`. Ces références seront remplacées par le texte capturé par la *n*-ième parenthèse capturante du masque. *n* peut prendre des valeurs de 0 à 99, et `\0` correspond au texte de qui satisfait le masque complet. Les parenthèses ouvrantes sont comptées de gauche à droit (en commençant à 1) pour déterminer le numéro de parenthèse capturante.

Si la recherche n'aboutit à aucun résultat, *subject* sera inchangé.

Tous les paramètres de [preg_replace\(\)](#) peuvent être des tableaux :

Si *subject* est un tableau, alors l'opération sera appliquée à chacun des éléments du tableau, et le tableau sera retourné.

Si *pattern* et *replacement* sont des tableaux, alors [preg_replace\(\)](#) prend une valeur de chaque tableau, et l'utilise pour faire la recherche et le remplacement. Si *replacement* a moins d'éléments que *pattern*, alors la chaîne vide est utilisé pour le reste des valeurs. Si *pattern* est un tableau, et que *replacement* est une chaîne, alors cette chaîne sera utilisée pour chaque valeur de *pattern*. Le contraire n'aurait pas de sens.

`/e` force [preg_replace\(\)](#) à traiter *replacement* comme du code PHP une fois que les substitutions adéquates ont été faites. Conseil : assurez vous que *replacement* est un code PHP valide, car sinon, PHP trouvera une erreur d'analyse (parse error) dans cette ligne. Note : *Cette option a été ajoutée en PHP 4.0.*

Remplacement de plusieurs valeurs

```
$patterns = array("/(19|20\d{2})-(\d{1,2})-(\d{1,2})/", "/^\s*{(\w+)}\s*="/);
$replace = array("\3/\4/\1", "$\1 =");
print preg_replace($patterns, $replace, "{startDate} = 1999-5-27");
```

Cet exemple va afficher :

```
$startDate = 5/27/1999
```

Utilisation de l'option /e

```
preg_replace("/(</?)(\w+)([>]*>)/e", "'\1'.strtoupper('\2').'\3'", $html_body);
```

Cela va mettre en majuscule toutes les balises HTML du texte.

Voir aussi [preg_match\(\)](#), [preg_match_all\(\)](#) et [preg_split\(\)](#).

[10.51.4 preg_split](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [preg_split](#) (string *pattern*, string *subject*, int *limit*, int *flags*)

Note : Le paramètre *flags* a été ajouté dans PHP Beta 3.

Retourne un tableau contenant les sous chaînes de *subject*, séparées par les chaînes qui vérifient *pattern*.

Si *limit* est donné, seules les *limit* premières chaînes seront retournées.

Si le flag est PREG_SPLIT_NO_EMPTY, alors seul les sous chaînes non nulles seront retournées.

Eclatement d'une chaîne de recherche.

```
$keywords = preg_split("/[\s,]+/", "hypertext language, programming");
```

Voir aussi [preg_match\(\)](#), [preg_match_all\(\)](#) et [preg_replace\(\)](#).

[10.51.5 preg_quote](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [preg_quote](#) (string *str*)

[preg_quote\(\)](#) ajoute un backslash devant tous les caractères de la chaîne *str*. Cela est très utile si vous avez une chaîne qui va servir de masque, mais qui est générée durant l'exécution.

Les caractères spéciaux qui seront échappés :

```
. \ + * ? [ ^ ] $ ( ) { } = ! < > | :
```

Note : Cette fonction a été ajoutée dans PHP 3.0.9.

[10.51.6 preg_grep](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [preg_grep](#) (string *pattern*, array *input*)

[preg_grep\(\)](#) retourne un tableau qui contient les éléments de *input* qui satisfont le masque *pattern*.

Exemple avec [preg_grep\(\)](#)

```
preg_grep("/^(\d+)?\.\d+$/", $array); // recherche les nombres à virgule flottante
```

Note : Cette fonction a été ajoutée dans PHP 4.0.

10.51.7 Options de recherche

[\[Notes en ligne\]](#)

Les options de PCRE sont listées ci dessous. Les noms entre parenthèses sont les noms internes à PCRE.

i (PCRE_CASELESS)

- Effectue une recherche insensible à la casse.

m (PCRE_MULTILINE)

- Par défaut, PCRE traite la chaîne sujet comme une seule ligne (même si cette chaîne contient des retours chariot). Le méta-caractère "début de ligne" (^) ne sera valable qu'une seule fois, au début de la ligne, et le méta caractère "fin de ligne" (\$) ne sera valable qu'à la fin de la chaîne, ou avant le retour chariot final (à moins que l'option E ne soit mise). C'est le même fonctionnement qu'en Perl. Lorsque cette option est mise, " début de ligne " et " fin de ligne " correspondront alors aux caractères suivant et précédent immédiatement un caractère de nouvelle ligne, en plus du début et de la fin de la chaîne. C'est le même fonctionnement que l'option Perl /m. Si il n'y a pas de caractère de nouvelle ligne "\n" dans la chaîne sujet, ou si il n'y a aucune occurrence de ^ ou \$ dans le masque, cette option ne sert à rien.

s (PCRE_DOTALL)

- Avec cette option, le méta caractère point (.) remplace n'importe quel caractère, y compris les nouvelles lignes. Sans cette option, le caractère point ne remplace pas les nouvelles lignes. Cette option est équivalente à l'option Perl /s. Une classe de caractère négative telle que [^a] acceptera toujours les caractères de nouvelles lignes, indépendamment de cette option.

x (PCRE_EXTENDED)

- Avec cette option, les caractères d'espacement sont ignorés, sauf lorsqu'ils sont échappés, ou à l'intérieur d'une classe de caractère, et tous les caractères entre # non échappés et en dehors d'une classe de caractère, et le prochain caractère de nouvelle ligne sont ignorés. C'est l'équivalent Perl de l'option /x : elle permet l'ajout de commentaires dans les masques compliqués. Notez bien, cependant, que cela ne s'appliquent qu'aux caractères de données. Les caractères d'espacement ne doivent jamais apparaître dans les séquences spéciales d'un masque, comme par exemple dans la séquence (? qui introduit une parenthèse conditionnelle.

e

- Avec cette option, [preg_replace\(\)](#) effectue la substitution normale des \\ dans la chaîne de remplacement, puis l'évalue comme un code PHP, et utilise le résultat pour remplacer la chaîne de recherche.

Seule [preg_replace\(\)](#) utilise cette option. Elle est ignorée par les autres.

Note : *Cette option a été ajoutée dans PHP 4.0.*

A (PCRE_ANCHORED)

- Avec cette option, le masque est ancré de force, c'est à dire que le masque doit s'appliquer entre le début et la fin de la chaîne sujet pour être considéré comme trouvé. Il est possible de réaliser le même effet en ajoutant les méta-caractères adéquats, ce qui est la seule manière de le faire en Perl.

E (PCRE_DOLLAR_ENDONLY)

- Avec cette option, le méta-caractère \$ ne sera valable qu'à la fin de la chaîne sujet. Sans cette option, \$ est aussi valable avant une nouvelle ligne, si cette dernière est le dernier caractère de la chaîne. Cette option est ignorée si l'option *m* est mise. Il n'y a pas d'équivalent en Perl.

S

- Lorsqu'un masque est utilisé plusieurs fois, cela vaut la peine de passer quelques instants de plus pour l'analyser et optimiser le code pour accélérer les traitements ultérieurs. Cette option force cette analyse plus poussée. Actuellement, cette analyse n'est utile que pour les masques non ancrés, qui ne

commencent pas par un caractère fixe.

U (PCRE_UNGREEDY)

- Cette option inverse la tendance à la gourmandise des expressions régulières. Vous pouvez aussi inverser cette tendance au coup par coup avec un `?`. De même, si cette option est mise, le `?` rendra gourmand une séquence. Cette option n'est pas compatible avec Perl. Elle peut aussi être mise dans le masque avec l'option `(?U)`.

X (PCRE_EXTRA)

- Cette option ajoute d'autres fonctionnalités incompatible avec le PCRE de Perl. Tous les backslash suivis d'une lettre qui n'aurai pas de signification particulière cause une erreur, permettant la réservation de ces combinaisons pour des ajouts fonctionnels ultérieurs. Par défaut, comme en Perl, les backslash suivi d'une lettre sans signification particulière sont traités comme des valeurs littérales. Actuellement, cette option ne déclenche pas d'autres fonctions.

[10.51.8 Syntaxe des masques](#)

[\[Notes en ligne\]](#)

La bibliothèque PCRE est un ensemble de fonctions qui implémentent la recherche par expressions régulières, en utilisant la même syntaxe et la même sémantique que le Perl 5, avec quelques nuances (voir ci-dessous). L'implémentation actuelle est celle de Perl 5.005.

Les différences avec le Perl 5.005 sont présentée ici :

1. Par défaut, un caractère d'espace correspond à n'importe quel caractère que la fonction C `isspace()` reconnaît, bien qu'il soit possible de recompiler la bibliothèque PCRE avec d'autres tables de caractères. Normalement, `isspace()` retourne TRUE pour les espaces, les retours chariot, les nouvelles lignes, les formfeed, les tabulations verticales et horizontales. Le Perl 5 n'accepte plus la tabulation verticale comme caractère d'espace. La séquence `\v` qui était dans la documentation Perl depuis longtemps n'a jamais été reconnue. Cependant, la tabulation verticale elle-même était reconnue comme un caractère d'espace jusqu'à la version 5.002. Avec les version 5.004 et 5.005, l'option `\s` l'ignore.
2. PCRE ne tolère pas la répétition de quantificateurs dans les expressions. Perl le permet, mais cela ne signifie pas ce que vous pourriez penser. Par exemple, `(?!a){3}` ne s'interprète pas : les trois caractères suivants ne sont pas des "a". En fait, cela s'interprète comme : le caractère suivant n'est pas "a" trois fois.
3. Les occurrences de sous-masques qui interviennent dans des assertions négatives sont comptées, mais elles ne sont pas enregistrées dans le vecteur d'occurrences. Perl modifie ses variables numériques pour toutes les occurrences de sous masque, avant que l'assertion ne vérifie le masque entier, et uniquement si les sous masques ne trouvent qu'une seule occurrence.
4. Bien que les caractères nul soient tolérés dans la chaîne de recherche, ils ne sont pas acceptés dans le masque, car le masque est utilisé comme une chaîne C standard, terminée par le caractère nul. Il faut donc utiliser la séquence d'échappement `"\0"` dans le masque pour rechercher les caractères nul.
5. Les séquence d'échappement suivantes ne sont pas supportées par le Perl: `\l`, `\u`, `\L`, `\U`, `\E`, `\Q`. En fait, elles sont implémentées par la gestion intrinsèque de chaînes du Perl, et ne font pas partie de ses caractères spéciaux.
6. L'assertion `\G` du Perl n'est pas supportée car elle n'est pas pertinente pour faire des recherches avec des masques uniques.
7. De manière assez évidente, PCRE n'accepte pas la construction `(?{code})`
8. Au moment de l'écriture de PCRE, Perl 5.005_02 avait quelques comportement étranges avec la capture des chaînes lorsqu'une partie du masque est redoublée. Par exemple, "aba" avec le masque `/^(a(b)?)+$/` va affecter à `$2` la valeur "b",

mais la même manipulation avec "aabbaa" et `/^(aa(bb)?)+$/` laissera \$2 vide. Cependant, si le masque est remplacé par `/^(aa(b(b))?) +$/` alors \$2 (et d'ailleurs \$3) seront correctement affectés. Avec le Perl 5.004, \$2 sera correctement affecté dans les deux cas, et c'est aussi vrai avec PCRE. Si Perl évolue vers un autre comportement cohérent, PCRE s'adaptera probablement.

9. Une autre différence encore non résolue est le fait qu'en Perl 5.005_02 le masque `/^(a)?(?1)a|b)+$/` accepte la chaîne "a", tandis que PCRE ne l'accepte pas. Cependant, que ce soit avec Perl ou PCRE `/^(a)?a/` et "a" laisseront \$1 vide.

10. PCRE propose quelques extensions aux expressions régulières du Perl.

(a) Bien que les assertions avec retour (lookbehind) soit obligée d'apparier une chaîne de longueur fixe, toutes les assertions avec retour peuvent avoir une longueur différente. Perl 5.005 leur impose d'avoir toutes la même longueur.

(b) Si PCRE_DOLLAR_ENDONLY est mis, et que PCRE_MULTILINE n'est pas mis, le méta caractère \$ ne s'applique qu'à la fin physique de la chaîne, et non pas avant les caractères de nouvelle ligne.

(c) Si PCRE_EXTRA est mis, un backslash suivi d'une lettre sans signification spéciale est considérée comme une erreur.

(d) SI PCRE_UNGREEDY est mis, la "gourmandise" des quantificateurs de répétition est inversées, ce qui est rend non gourmand par défaut, mais si ils sont suivis de ?, il seront gourmands.

La syntaxe et la sémantique des expressions régulière supportées par PCRE sont décrites ci-dessous. Les expressions régulières sont aussi décrites dans la documentation Perl, et dans un grand nombre d'autres livres, avec de nombreux exemples. Jeffrey Friedl's "Mastering Regular Expressions", édité chez O'Reilly

(ISBN 1-56592-257-3), les décrits en profondeur. Cette description est organisée comme une documentation de référence.

Une expression régulière est un masque, qui est appliqué sur une chaîne sujet, de gauche à droite. La plus part des caractères se représentent eux-mêmes. Un exemple trivial : un masque qui serait

Le rapide renard gris

Pourra correspondre à une partie de la chaîne sujet qui sera identique au masque.

La puissance des expressions régulières provient de leur capacité à autoriser des alternatives et des quantificateur de répétitions dans le masque. Ils sont encodés dans le masque par des *meta-characters*,

qui ne représentent pas ce qu'ils sont, mais sont interprétés d'une certaine manière.

Il y a deux sortes de méta-caractères : ceux qui sont reconnus n'importe où dans un masque, hormis entre crochets, et ceux qui sont reconnus entre crochets. A l'extérieur des crochets, les méta caractères sont :

<code>\</code>	Caractère d'échappement, avec de multiples usages.
<code>^</code>	Le début de la chaîne sujet (ou de ligne, en mode multiligne)
<code>\$</code>	La fin de la chaîne sujet (ou de ligne, en mode multiligne)
<code>.</code>	Remplace n'importe quel caractère, hormis le caractère de nouvelle ligne (par défaut) ;
<code>[</code>	Caractère de début de définition de classe
<code>]</code>	Caractère de fin de définition de classe
<code> </code>	Caractère de début d'alternative
<code>(</code>	Caractère de début de sous masque
<code>)</code>	Caractère de fin de sous masque
<code>?</code>	Etend le sens de (

mais aussi quantificateur de 0 ou 1

mais aussi quantificateur de minimisation

<code>*</code>	Quantificateur de 0 ou plus
<code>+</code>	Quantificateur de 1 ou plus
<code>{</code>	Caractère de début de quantificateur minimum/maximum

La partie du masque qui est entourée de crochet et appelé une classe de caractères. Dans les clas

<code>\</code>	Caractère d'échappement, avec de multiples usages
<code>^</code>	négation de la classe, mais uniquement si placé tout au début de la classe
<code>-</code>	indique un intervalle de caractères

] termine la classe de caractères

La section suivante décrit l'utilisation de chaque méta caractères :

BACKSLASH

Le caractère backslash a de nombreuses utilisations. En premier lieu, si il est suivi d'un caractère non alpha numérique, il ne prendra pas la signification spéciale qui y est rattachée. Cette utilisation du backslash comme caractère d'échappement s'applique à l'intérieur et à l'extérieur des classes de caractères.

Par exemple, pour recherche le caractère étoile "*", il faut écrire dans le masque : "*". Cela s'applique dans tous les cas, que le caractère qui suive soit un méta-caractère ou non. C'est un moyen sûr pour s'assurer qu'un caractère sera recherché pour sa valeur littérale, plutôt que pour sa valeur spéciale. En particulier, pour rechercher les backslash, il faut écrire : "\\".

Si un masque est utilisé avec l'option PCRE_EXTENDED, les espaces blancs du masque, mais qui ne sont pas dans une classe de caractères, et les caractères entre dièses "#", ainsi que les nouvelles lignes sont ignorées. Le backslash peut être utilisé pour échapper et ainsi rechercher un espace ou un dièse.

La deuxième utilité du backslash est de pouvoir coder des caractères invisibles dans les masques. Il n'y a pas de restriction sur la place de ces caractères invisibles, hormis pour le caractère nul qui doit terminer le masque. Lors de la préparation du masque, il est souvent plus pratique d'utiliser les séquences d'échappement suivantes, plutôt que le caractère binaire qu'elle représente :

```
\a      alarme, c'est à dire le caractère BEL (hex 07)
\cx     "control-x", avec x qui peut être n'importe quel caractère.
\e      escape (hex 1B)
\f      formfeed (hex 0C)
\n      nouvelle ligne (hex 0A)
\r      retour chariot (hex 0D)
\t      tabulation (hex 09)
\xhh    caractère en hexadécimal, de code hh
\ddd    caractère en octal, de code ddd, ou référence arrière
```

Dans la séquence "\cx" si "x" est en minuscule, il est converti en majuscule. Puis, le bit 6 (hex 40) est inversé. Ainsi "\cz" devient 1A, mais "\c{" devient hex 3B, tandis que "\c;" devient hex 7B.

Après "\x", deux caractères hexadécimaux sont lus (les lettres peuvent être en majuscule ou minuscule).

Après "\0", deux caractères octal sont lus. Dans chacun des cas, le méta-caractère tente de lire autant de caractère que possible. Ainsi la séquence

```
"\0\x\07", sera comprise comme deux caractères nuls, suivi d'un caractère alarme (BEL). Assurez vous que vous fournissez suffisamment de chiffres après le
```

méta-caractère.

La gestion de la séquence "\y", avec y <> 0 est plutôt compliquée. En dehors des caractères de classes, PCRE va lire y et tous les caractères qui suivent comme des chiffres décimaux. Si y est plus petit que 10, ou bien si il y a déjà eu au moins autant de parenthèses ouvrantes auparavant, la séquence est prise pour une *référence de retour*. Le détail sera vu ultérieurement, après la section sur les sous-masques.

A l'intérieur d'un caractère de classe, ou si y est plus grand que 10, et qu'il n'y a pas eu assez de parenthèses ouvrantes auparavant, PCRE lis jusqu'à 3 chiffres octals à la suite du backslash, et génère un octet unique, à partir des 8 bits de poids faible de la séquence. Tous les chiffres qui suivent ne sont pas interprétés, et se représentent eux-mêmes. Par exemple,

```
\040    est une autre manière d'écrire un espace
\40      est identique, dans la mesure où il n'y a pas 40 parenthèses
```

ouvrantes auparavant.

```
\7       est toujours une référence de retour.
\11      peut être une référence de retour, ou une tabulation, tandis que
\011     est toujours une tabulation
```

```
\011     est toujours une tabulation
\0113    est une tabulation suivi du caractère "3"
```

\113 est le caractère 113 (étant donné qu'il ne peut y avoir plus de 99 référence de arrière)

```
\377    est un octet dont tous les bits sont à 1
```

`\81` peut être soit une référence de arrière, soit le caractère NULL, suivi des caractères "8" et "1"

Les valeurs octales supérieures ou égales à 100 ne doivent pas être introduite par un 0, car seuls les trois premiers octets seront lus.

Toutes les séquences qui définissent une valeur d'un seul octet peuvent être utilisé dans les classes de caractères, et à l'extérieur. De plus, dans une classe de caractère, la séquence `"\b"` est interprétée comme un caractère effacer

(backspace, hex 08). A l'extérieur d'une classe de caractères, il peut avoir d'autres significations (voir ci-dessous).

On peut encore se servir de backslash pour préciser des types génériques de valeurs :

```
\d    tout caractère décimal
\D    tout caractère qui n'est pas un caractère décimal
\s    tout caractère blanc
\S    tout caractère qui n'est pas un caractère blanc
\w    tout caractère de "mot"
\W    tout caractère qui n'est pas un caractère de "mot"
```

Chaque paire précédente définit une partition de la table des caractères : les deux ensembles sont disjoints. Un caractère satisfera soit un méta-caractère, soit l'autre.

Un caractère de "mot" sera une lettre, un chiffre ou le caractère souligné, c'est à dire un caractère qui pourra être une partie d'un mot Perl. La définition des lettres et chiffres est définie par les tables de caractères de PCRE, et peut varier suivant la table locale de caractère (voir "Tables de caractères locales ", ci-dessus. Par exemple, dans la configuration français ("fr"), certains caractères ont des codes supérieurs à 128, pour les caractères accentués, et ils seront compris par le méta caractère `\w`.

Ces séquences de caractères peuvent apparaître à l'intérieur ou à l'extérieur des classes de caractères. Elles remplacent à chaque fois un caractère du type correspondant. Si cette séquence est mis en fin de masque, et qu'il n'y a plus de caractère à comparer dans la chaîne sujet, la recherche échoue.

La quatrième utilisation du backslash intervient lors d'assertions simples.

Une assertion impose une condition à un certain point, sans remplacer de caractère. L'utilisation de sous-masques pour réaliser des assertions plus complexes est décrites plus bas. Les assertions avec backslash sont les suivantes :

```
\b    limite de mot
\B    pas limite de mot
\A    début de la chaîne sujet (indépendant du mode multi-lignes)
\Z    fin de la chaîne sujet ou nouvelle ligne à la fin de la chaîne sujet
      (indépendant du mode multi-lignes)
\z    fin de la chaîne sujet (indépendant du mode multi-lignes)
```

Ces assertions ne peuvent pas apparaître dans une classe de caractère

(mais `"\b"` a une autre signification à l'intérieur d'une classe de caractères).

Une limite de mot est un emplacement dans la chaîne sujet ou un caractère et son suivant ne sont pas en même temps des caractères de mot, ou le contraire

(on peut le voir comme `\w\W` ou `\W\w`), ou encore le premier ou le dernier caractère est un caractère mot.

Les assertions `\A`, `\Z`, et `\z` diffèrent des méta caractères `^` et `$` dans la mesure où ils ne sont pas dépendants des options, notamment `PCRE_NOTBOL` ou `PCRE_NOTEOL`.

La différence entre `\Z` et `\z` tient au fait que `\Z` recherche les positions avant les nouvelles lignes et à la fin de la chaîne sujet, tandis que `\z` ne recherche que la fin de la chaîne.

CIRCUMFLEX et DOLLAR

En dehors d'une classe de caractère, avec les options par défaut,

`^` est une assertion qui n'est vraie que si elle est placée tout au début de la chaîne. A l'intérieur d'une classe de caractère, `^` a un tout autre sens (voir ci-dessous).

`^` n'a pas besoin d'être le premier caractère du masque, si plusieurs alternatives sont proposées, mais il doit être placé en premier dans chaque alternative.

Si toutes les alternatives commencent par `^`, alors le masque est dit ancré

(il y a une autre construction qui porte cette appellation).

\$ est une assertion qui n'est vraie que si elle est placée tout en fin de chaîne ou juste avant un caractère de nouvelle ligne qui serait le dernier caractère de la chaîne. A l'intérieur d'une classe de caractère, ^ a un tout autre sens (voir ci-dessous).

\$ n'a pas besoin d'être le dernier caractère du masque, si plusieurs alternatives sont proposées, mais il doit être placé en dernier dans chaque alternative. Si toutes les alternatives finissent par \$, alors le masque est dit ancré (il y a une autre construction qui porte cette appellation). \$ n'a pas de valeur particulière dans une classe de caractères.

La signification de \$ peut changer, de manière à l'amener à ce qu'il ne puisse se trouver qu'en toute fin de la chaîne sujet. Cela se fait en ajoutant l'option PCRE_DOLLAR_ENDONLY au moment de la compilation, ou de l'exécution. Cette option est inopérante sur \Z.

La signification de ^ peut changer, de manière à l'amener à ce qu'il puisse se trouver immédiatement avant et immédiatement après un caractère de nouvelle ligne

"\n". Cela se fait en ajoutant l'option PCRE_MULTILINE au moment de la compilation, ou de l'exécution. Par exemple, le masque /^abc\$/ accepte la chaîne "def\nabc" uniquement en mode multi-lignes. Par conséquent, toutes les parties du masques qui commencent par "^" ne sont pas ancrées, en mode multi ligne. L'option PCRE_DOLLAR_ENDONLY est ignorée si l'option PCRE_MULTILINE est choisie.

Notez que les méta caractères \A, \Z, et \z peuvent servir à repérer le début et la fin du sujet, et toutes les parties du masque qui commenceront par \A seront toujours ancrées, avec l'option PCRE_MULTILINE ou non.

FULL STOP (PERIOD, DOT)

En dehors d'une classe de caractères, un point remplace n'importe quel caractère, même invisible et à l'exception du caractère de nouvelle ligne. Avec l'option PCRE_DOTALL le point remplace n'importe quel caractère, même le caractère de nouvelle ligne. La gestion des points est complètement indépendante de ^ et \$. Le seul point commun est que les deux ont un comportement particulier vis à vis des caractère de nouvelle ligne. Le point n'a pas de comportement particulier dans une classe de caractères.

SQUARE BRACKETS

Un crochet ouvrant introduit une classe de caractère, et le crochet fermant la conclut. Le crochet fermant n'a pas de signification en lui même. Si le crochet fermant est nécessaire à l'intérieur d'une classe de caractères, il faut qu'il soit le premier caractère (après un ^ éventuel) ou échappé avec un backslash. Une classe de caractère remplace un seul caractère dans la chaîne sujet, à moins que le premier caractère de la classe soit un ^, qui représente une négation : le caractère ne doit pas se trouver dans la classe. Si ^ est nécessaire dans la classe, il suffit qu'il ne soit pas le premier caractère, ou bien qu'il soit échappé avec un backslash.

Par exemple, le caractère [aeiou] remplace n'importe quelle voyelle minuscule, tandis que [^aeiou] remplace n'importe quelle caractère qui n'est pas une voyelle minuscule. ^ est une notation pratique pour spécifier des caractères qui sont dans une classe, en ne citant que ceux qui n'y sont pas. Le comportement est inchangé.

Avec l'option d'insensibilité à la casse, toutes les lettres d'une classe de caractère représentent en même temps la majuscule et la minuscule. Par exemple, [aeiou] représentera "A" ou "a", et [^aeiou] n'acceptera pas "A", tandis que sans l'option, elle l'accepterait.

Le caractère de nouvelle ligne n'est pas traité de manière spéciale dans les classes de caractère, quelque soit l'option PCRE_DOTALL ou PCRE_MULTILINE.

Une classe telle que [^a] acceptera toujours une nouvelle ligne.

Le signe moins (-) est utilisé pour spécifier un intervalle de caractères, dans une classe. Par exemple, [d-m] remplace toutes les lettres entre d et m inclus. Si le caractère moins est requis dans une classe, il faut l'échapper avec un backslash, ou le faire apparaître à une position où il ne pourra pas être interprété comme une indication d'intervalle, c'est à dire au début ou à la fin de la classe.

Il n'est pas possible d'avoir le caractère "]" comme fin d'intervalle. Un masque tel que [W-]46] est compris comme la classe de caractère contenant deux caractères

("W" et "-") suivi de la chaîne littérale "46]", ce qui fait qu'il va accepter

"W46]" ou "-46]". Cependant, si "]" est échappé avec un backslash, le masque [W-\]46] est interprété comme une classe d'un seul caractère, contenant un intervalle de caractère. La valeur octale ou hexadécimale de "]" peuvent aussi être utilisée pour déterminer les limites de l'intervalle. Les intervalles travaillent sur des séquences ASCII. Elles peuvent aussi être précisées avec des valeurs numériques, par exemple [\000-\037]. Si cet intervalle inclus des lettres utilisées avec une option d'insensibilité de casse, les majuscules ou minuscules correspondantes seront aussi incluses. Par exemple, [W-c] est équivalent à [][\^_`wxyzabc], avec l'option d'insensibilité de casse. Si la table locale de caractère est "fr", [\xc8-\xcb] correspond aux caractères accentués. Les types de caractères \d, \D, \s, \S, \w, et \W peuvent aussi intervenir dans les classes de caractères. Par exemple, [\dABCDEF] acceptera n'importe quel caractère hexadécimal. Un accent circonflexe peut aussi être utilisé pour spécifier adroitement des ensembles de caractères plus restrictifs : par exemple [^\W_] accepte toutes les lettres et les chiffres, mais pas les soulignés. Tous les caractères non alphanumériques autres que \, -, ^ (placé en début de chaîne) et] n'ont pas de significations particulière, mais ils ne perdront rien à être échappés.

VERTICAL BAR

La barre verticale sert à séparer des alternatives. Par exemple, dans le masque dupont|martin recherche soit "dupont", soit " martin ". Le nombre d'alternative n'est pas limité, et il est même possible d'utiliser la chaîne vide. Lors de la recherche, toutes les alternatives sont essayées, de gauche à droite, et la première qui est acceptée, est utilisée. Si les alternatives sont dans un sous masque, elle ne réussiront que si le masque principal réussi aussi.

INTERNAL OPTION SETTING

Les options PCRE_CASELESS, PCRE_MULTILINE, PCRE_DOTALL, et PCRE_EXTENDED peuvent être changée depuis le masque lui-même, avec des séquences mises "(?" et ")".

Les options sont

i pour PCRE_CASELESS
m pour PCRE_MULTILINE
s pour PCRE_DOTALL
x pour PCRE_EXTENDED

Par exemple, (?im) rend le masque insensible à la casse, et multi-lignes. Il est possible d'annuler ces options en les faisant précéder par un signe - : par exemple (?im-sx), ajoutera les options PCRE_CASELESS et PCRE_MULTILINE mais annulera les options PCRE_DOTALL et PCRE_EXTENDED. Si une option apparaît avant et après le signe moins, l'option sera annulée.

Le domaine d'application de ces options dépend de la position de la séquence d'option. Pour toutes les séquences d'options qui sont hors des sous masques (définis plus loin), l'effet est le même que si l'option avait été fixée dès le début de la recherche. Les exemples suivants se comportent tous de la même façon :

```
(?i)abc
a(?i)bc
ab(?i)c
abc(?i)
```

et sont parfaitement équivalents au masque abc avec l'option PCRE_CASELESS.

En d'autres termes, mettre des séquences d'options dans le corps principal du masque revient à appliquer l'option à tout le masque, sauf ordre contraire dans les sous masques. Si il y a plusieurs séquences d'option qui portent sur la même optin, la dernière s'appliquera.

Si une option intervient dans un sous-masque, le comportement est différent.

C'est un changement de comportement apparu en Perl 5.005. Une option à l'intérieur d'un sous masque n'affecte que cette partie du masque, ce qui fait que

```
(a(?i)b)c
```

acceptera abc et aBc mais aucune autre chaîne (en supposant que PCRE_CASELESS n'est pas utilisé). Cela signifie que les options permettent d'avoir différente configuration de recherche pour différentes parties du masque. Une séquence d'option dans une alternative affecte toute l'alternative. Par exemple :

`(a(?i)b|c)`

accepte "ab", "aB", "c", et "C", même si, comme dans le cas de "C", la première alternative qui porte l'option n'est pas prise en compte. Sinon, cela risque d'introduire des comportements très étranges :

Les options spécifiques à PCRE telles que PCRE_UNGREEDY et PCRE_EXTRA peuvent être modifiées de la même manière, en utilisant respectivement les caractères U et X. L'option (?X) est particulière, car elle doit toujours intervenir avant toutes les autres options, même au niveau du masque entier. Il vaut mieux la mettre au début du masque.

SUBPATTERNS

Les sous masques sont délimités par des parenthèses, et peuvent être imbriqués. Ajouter des sous masques a deux utilités :

1. Délimiter des alternatives. Par exemple, le masque

`cat(aract|erpillar|)`

acceptera les mots "char", "chardon", ou "charmant". Sans les parenthèses, il n'accepterait que "chardon", "mant" ou la chaîne vide.

2. Le sous masque est considéré comme capturant : Lorsqu'une chaîne sujet est acceptée par le masque, le sous masque est capturé. Par exemple, soit la chaîne sujet "le roi soleil " qui est utilisée avec le masque suivant :

Le `((roi|prince) (soleil|charmant))`

les sous masques capturés sont "roi soleil ", "roi", et "soleil", numérotés respectivement 1, 2, et 3.

L'ubiquité des parenthèses n'est pas toujours simple d'emploi. Il y a des moments où regrouper des sous masques est nécessaire, sans pour autant capturer la valeur trouvée. Si une parenthèse ouvrante est suivie de "?:", le sous masque ne capture pas la chaîne assortie, et ne sera pas compté lors de la numérotation des captures. Par exemple, avec la chaîne "le prince charmant", utilisé avec le masque pattern

Le `((?roi|prince) (soleil|charmant))`

les chaînes capturées seront "prince charmant " et "charmant", numérotés respectivement 1 et 2. Le nombre maximal de chaîne capturées est de 99, et le nombre total de sous masque (capturant ou non) ne doit pas dépasser 200.

`(?i:samedi|dimanche)`

`(?:(?i) samedi | dimanche)`

De plus, comme les séquences d'options sont valables sur toute une alternative, le masque ci dessus acceptera aussi "DIMANCHE" que "Dimanche".

REPETITION

les Répétitions sont spécifiées avec des quantificateurs, qui peuvent être placés à la suite des caractères suivants :

Un caractère unique, même s'il s'agit d'un méta caractère

Une classe de caractères

Une référence de retour (Voir section suivante)

Un sous masque avec parenthèses (a moins que ce ne soit une assertion, voir plus loin)

Les quantificateurs généraux précisent un nombre minimum et maximum de répétitions possibles, donnés par deux nombres entre accolades, et séparés par une virgule.

Ces nombres doivent être plus petit que 65536, et le premier nombre doit être égal ou inférieur au second. Par exemple

`z{2,4}`

accepte "zz", "zzz", ou "zzzz". L'accolade fermante n'a pas de signification par elle même. Si le second nombre est omis, mais que la virgule est là, cela signifie qu'il n'y a pas de limite supérieure. Si le second nombre et la virgule sont omis, le quantificateur correspond au nombre exact de répétition attendues.

Par exemple :

`[aeiou]{3,}`

accepte n'importe quelle succession d'au moins 3 voyelles minuscules, tandis que

`\d{8}`

n'accepte que 8 chiffres exactements. Une accolade ouvrante qui apparaît à une position où un quantificateur n'est pas accepté, ou si la syntaxe des quantificateurs n'est pas respectée, elle sera considérée littérale. Par exemple,

`{,6}` n'est pas un quantificateur, mais une chaîne de 4 caractères.

Le quantificateur {0} est autorisé, mais l'expression est alors ignorée.

* équivalent à {0,}

+ équivalent à {1,}

? équivalent à {0,1}

Il est possible de constituer des boucles infinies en créant un sous masque sans caractères, mais pourvu d'un quantificateur sans limite supérieure.

Par exemple

```
(a?)*
```

Les versions plus anciennes de Perl et PCRE généraient alors une erreur au moment de la compilation. Cependant, étant donné qu'il existe des situations où ces constructions peuvent être utiles, ces masques sont désormais autorisés. Cependant, si la répétition du sous masque ne trouve aucun caractère, la boucle est interrompue.

Par défaut, les quantificateurs sont "gourmands", c'est à dire, qu'ils cherchent d'abord à trouver le nombre maximal de répétitions qui autorise le succès de la recherche. L'exemple classique posé par cette gourmandise est la recherche de commentaire d'un programme en C. Les commentaires apparaissent entre les séquences /* et */ et à l'intérieur de ces délimiteurs, les * et / sont autorisés.

Appliquer le masque

```
/\*.*\*/
```

à la chaîne

```
/* first comment */ not comment /* second comment */
```

ne peut réussir, car le masque travaille sur toute la chaîne, à cause de la gourmandise du caractère .*.

Cependant, un quantificateur suivi d'un point d'interrogation cesse d'être gourmand, et au contraire, ne recherche que le nombre minimum de répétition.

Dans ces conditions, le masque

```
/\*.*?\*/
```

trouvera bien les commentaires du code C. La signification des autres quantificateurs n'est pas changée. Attention à ne pas confondre l'utilisation du point d'interrogation ici avec son utilisation comme quantificateur lui même. A cause cette ambiguïté, il peut apparaître des situations où il faut le doubler :

```
\d??\d
```

Ce masque va tenter de lire un seul chiffre, mais le cas échéant, il acceptera 2 chiffres pour permettre à la recherche d'aboutir.

Si l'option PCRE_UNGREEDY est mise, (une option qui n'est pas disponible avec Perl) alors les quantificateurs sont non gourmand par défaut, mais peuvent être rendu gourmand au cas par cas, en ajoutant un point d'interrogation après. En d'autres termes, cette option inverse le comportement par défaut.

Lorsqu'un sous masque est quantifié avec un nombre minimum de répétition, qui soit plus grand que 1, ou avec un maximum de répétition, le masque compilé aura besoin de plus de place de stockage, proportionnellement au minimum et au maximum.

Si un masque commence par .* ou {0,} et que l'option PCRE_DOTALL (équivalent en Perl à /s) est mise, c'est à dire en autorisant le remplacement des nouvelles lignes par un méta caractère, alors le masque est implicitement ancré, car tout ce qui suit va être mangé par la première séquence, et se comportera comme si le masque se terminait par le méta caractère \A. Dans le cas où on sait d'avance qu'il n'y aura pas de caractère de nouvelle ligne, mettre l'option PCRE_DOTALL et commencer le masque par .* permet d'optimiser le masque. Alternativement, on peut utiliser

```
^ pour ancrer explicitement le masque.
```

Lorsqu'un sous masque capturant est répété, la valeur capturée est la dernière.

Par exemple, après que

```
(inter[net]{3}\s*)+
```

```
*)+
```

ai été appliqué à "internet interne" la valeur de la chaîne capturée est

"interne". Cependant, si il y a des sous masques imbriqués, la valeur capturée correspondante peut l'avoir été lors des précédentes itérations. Par exemple :

```
/(a|(b))+/
```

accepte "aba" et la deuxième valeur capturée est

Références arrières (back references)

En dehors des classes de caractères, un backslash suivi d'un nombre plus grand

que 0 (et possiblement plusieurs chiffres) est une référence arrière (c'est à dire vers la gauche) dans le masque, en supposant qu'il y ait suffisamment de sous masques capturant précédent.

Cependant, si le nombre décimal suivant le backslash est plus petit que 10, il sera toujours considéré comme une référence arrière, et cela générera une erreur si le nombre de capture n'est pas suffisant. En d'autres termes, il faut qu'il existe suffisamment de parenthèses ouvrantes à gauche de la référence, surtout si la référence est inférieure à 10. Reportez vous à la section "Backslash" pour avoir de plus amples détails à propos du nombre de chiffres qui suivent le backslash. La référence arrière remplace ce qui a été capturé par un sous masque dans le masque courant, plutôt que remplace le sous masque lui même. Ainsi

```
(calme|rapide) et \lment
```

trouvera "calme et calmement " et "rapide et rapidement ", mais pas

```
" calme et rapidement ". Si la recherche tiens compte de la casse, alors la casse de la chaîne capturée sera importante. Par exemple,
```

```
((?i)rah)\s+\l
```

trouve "rah rah" et "RAH RAH", mais pas "RAH rah", même si le sous masque capturant initial ne tenait pas compte de la casse.

Il peut y avoir plusieurs références arrières dans le même sous masque. Si un sous masque n'a pas été utilisé dans une recherche, alors les références arrières échoueront. Par exemple

```
(a|(bc))\2
```

ne réussira jamais si la chaîne sujet commence par "a" plutôt que par "bc".

Etant donné qu'il peut y avoir jusqu'à 99 références arrières, tous les chiffres après le backslash sont considérés comme faisant potentiellement partie de la référence arrière. Si le masque recherche un chiffre après la référence, alors il faut impérativement utiliser des délimiteurs pour terminer la référence arrière. Si l'option PCRE_EXTENDED est mise, on peut utiliser un espace. Sinon, un commentaire vide fait l'affaire.

Une référence arrière qui intervient à l'intérieur de parenthèses auquel elle fait référence échouera dès que le sous masque sera utilisé. Par exemple,

```
(a\1) échouera toujours. Cependant, ces références peuvent être utiles dans les sous masques répétitifs. Par exemple, le masque
```

```
(a|b\1)+
```

pourra convenir pour "a", "aba", "ababaa" etc. A chaque itération du sous masque, la référence arrière utilise le résultat du dernier sous masque. Pour que cela fonctionne, il faut que la première itération n'ai pas besoin d'utiliser la référence arrière. Cela arrive avec les alternatives, comme dans l'exemple ci dessus, ou avec un quantificateur de minimum 0.

Les assertions

Une assertion est un test sur les caractères suivants ou précédent celui qui est en cours d'étude. Ce test ne consomme pas de caractère (ie, on ne déplace pas le pointeur de caractères). Les assertions simples sont codées avec \b, \B, \A, \Z,

```
\z, ^ et $, et sont décrite précédemment. Il existe cependant un type
```

d'assertion plus complexe, codées sous la forme de sous masques. Il en existe deux types : ceux qui travaille au dela de la position courante, et ceux qui travaille en deça.

```
\w+(?=;)
```

Une assertion se comporte comme un sous masque, hormis le fait qu'elle ne déplace pas le pointeur de position. Les assertions avant commencent par (?= pour les assertions positives et par (?! pour des assertions négatives. Par exemple : foo(?!bar)

s'assure qu'un mot est suivi d'un point virgule, mais n'inclus pas le point virgule dans la capture. D'autre part,

```
(?!foo)bar
```

en est proche, mais ne trouve pas une occurrence de "bar" qui soit précédée par quelque chose d'autre que "foo"; il trouve toutes les occurrences de "bar", quelque soit ce qui le précède, car l'assertion (?!foo) est toujours vraie quand les trois caractères suivants sont "bar". Une assertion arrière est ici nécessaire. Ces assertions commencent par (?<= pour les assertions positives, et (?<! pour les assertions négatives. Par exemple :

```
(?<!foo)bar
```

trouve les occurrences de "bar" qui ne sont pas précédées par "foo". Le contenu d'une référence arrière est limité de tel façon que les chaînes qu'il utilise sont toujours de la même taille. Cependant, lorsqu'il y a plusieurs alternatives, elles n'ont pas besoin d'être de la même taille.

```
(?<=bullock|donkey)
```

est autorisé, tandis que

```
(?<!dogs?|cats?)
```

provoque une erreur de compilation. Les alternatives qui ont des longueurs différentes ne sont autorisées qu'au niveau supérieur des assertions arrières. C'est une amélioration du fonctionnement de Perl 5.005, qui impose aux alternatives d'avoir toutes la même taille. Une assertion telle que

```
(?<=ab(c|de))
```

n'est pas autorisée, car l'assertion de bas niveau (la deuxième, ici) a deux alternatives de longueurs différentes. Pour la rendre acceptable, il faut écrire

```
(?<=abc|abde)
```

L'implémentation des assertions arrières déplace temporairement le pointeur de position vers l'arrière, et cherche à vérifier l'assertion. Si le nombre de caractère est différent, la position ne sera pas correcte, et l'assertion

échouera. La combinaison d'assertions arrières avec des sous masques peut être particulièrement pratique à fin des chaînes. Un exemple est donné à la fin de cette section.

Plusieurs assertions peuvent intervenir successivement. Par exemple

```
(?<=\d{3})(?!999)foo
```

recherche les chaînes "foo" précédée par trois chiffres qui ne sont pas "999".

De plus, les assertions peuvent être imbriquées :

```
(?<=(?!foo)bar)baz
```

recherche les occurrences de "baz" qui sont précédées par "bar", qui, à son tour, n'est pas précédé par "foo".

Les assertions ne sont pas capturantes, et ne peuvent pas être répétées. Si une assertion contient des sous masques capturants en son sein, ils seront compris dans le nombre de sous masques capturants du masque entier. La capture est réalisée pour les assertions positives, mais cela n'a pas de sens pour les assertions négatives.

200 assertions au maximum sont autorisées.

Sous masques uniques (ONCE-ONLY SUBPATTERNS)

Avec les quantificateurs de répétitions, l'échec d'une recherche conduit normalement à une autre recherche, avec un nombre différent de répétitions, pour voir si le masque ne s'applique pas dans d'autres conditions. Parfois, il est pratique d'éviter ce comportement, soit pour changer la nature de la recherche, soit pour la faire abandonner plus tôt, si on pense qu'il n'est pas besoin d'aller plus loin.

Considérons par exemple, le masque `\d+foo` appliqué à la ligne

```
123456bar
```

Après avoir tenté d'utiliser les 6 chiffres suivi de "foo" qui fait échouer, l'action habituelle sera de réessayer avec 5 chiffres, puis avec 4, et ainsi de suite jusqu'à l'échec final. Un sous masque évalué une seule fois permettrait d'indiquer que lorsqu'une partie du masque est trouvée, elle n'a pas besoin d'être réévaluée à chaque tentative. Ceci conduirait à ce que la recherche

échoue immédiatement après le premier test. Ces assertions ont leur propre notation, commençant avec `(?>` comme ceci :

```
(?>\d+)bar
```

Après avoir tenté d'utiliser les 6 chiffres suivi de "foo" qui fait échouer, l'action habituelle sera de réessayer avec 5 chiffres, puis avec 4, et ainsi de suite jusqu'à l'échec final. Un sous masque évalué une seule fois permettrait d'indiquer que lorsqu'une partie du masque est trouvée, elle n'a pas besoin d'être réévaluée à chaque tentative. Ceci conduirait à ce que la recherche

échoue immédiatement après le premier test. Ces assertions ont leur propre notation, commençant avec `(?>` comme ceci :

```
(?>\d+)bar
```

Ce type de parenthèses verrouille le sous masque qu'il contient un fois qu'il a été trouvé, et empêche un échec ultérieur d'y repasser, mais autorise à revenir plus loin en arrière.

Une autre description est que les sous masques de ce type recherche les chaînes de caractères, et les ancre le sous masque à l'intérieur de la chaîne.

Les sous masques uniques ne sont pas capturants. Des cas simples comme ceux présentés ci dessus peuvent être pris comme des situations maximisantes, qui réservent le maximum de caractères. En effet, alors que `\d+` et `\d?` ajustent le nombre de chiffres trouvés de manière à laisser la possibilité au masque de réussir, `(?>\d+)` ne peut retenir que la séquence entière de chiffres.

Cette construction peut contenir un nombre arbitraire de sous masques complexes, et ils peuvent être imbriqués.

Les sous masques uniques ne peuvent être utilisés qu'avec les assertions arrières, pour effectuer une recherche efficace en fin de chaîne. Considérons un masque simple tel que

```
abcd$
```

appliqué à une très longue chaîne qui ne lui correspond pas. A cause du système de recherche de gauche à droite, PCRE va commencer par rechercher un "a" dans la chaîne sujet, puis vérifier si ce qui suit convient au reste du masque. Si le masque est spécifié sous la forme

```
^.*abcd$
```

alors, la séquence `.*` remplace en premier lieu la chaîne entière, et échoue, repart en arrière, et remplace tous les caractères sauf le dernier, échoue, retourne en arrière, prend un caractère de moins, etc... et ainsi de suite.

Encore une fois, la recherche du "a" passe en revue toute la chaîne de gauche

à droite, ce qui n'est pas très efficace. Par contre, si le masque était écrit

```
^(?>.*)(?<=abcd)
```

alors il n'y aurait pas de retour en arrière, pour satisfaire la séquence `.*`; car elle ne peut que remplacer toute la chaîne. L'assertion arrière consécutive va alors faire un test sur les 4 derniers caractères. Si elle échoue, la recherche est immédiatement interrompue. Pour les chaînes très longues, cette approche fait la différence en terme de performance et de temps de recherche.

Les sous masques conditionnels (CONDITIONAL SUBPATTERNS)

Il est possible de lier un sous masque à une condition, ou de choisir entre deux sous masques alternatifs, en fonction du résultat d'une assertion, ou suivant les résultats de recherche précédents. Les deux formes possibles de sous masques conditionnels sont

```
(?(condition)masque positif)
```

```
(?(condition) masque positif | masque négatif)
```

Si les conditions sont satisfaites, le masque positif est utilisé, sinon, le masque négatif est utilisé, si présent. Si il y a plus de deux alternatives, une erreur est générée à la compilation.

Il y a deux types de conditions. Si le texte entre les parenthèses est une séquence de chiffre, alors la condition est satisfaite si le sous masque correspondant à ce numéro a réussi. Considérons le masque suivant, qui contient des espaces non significatif pour le rendre plus compréhensible (on supposera l'option PCRE_EXTENDED mise) et qui est divisé en trois parties pour simplifier les explications

```
( \ ( )? [^()]+ (?(1) \ ) )
```

La première partie recherche une parenthèse ouvrante optionnelle, et si elle existe, elle est capturée. La deuxième partie recherche un séquence de caractères qui ne contient pas de parenthèses. La troisième partie est conditionnée à la première, et s'assure que si il y avait une parenthèse ouvrante, il en existe une fermante. Si une parenthèse ouvrante a été trouvée, elle a été capturée, et donc la première capture existe, et la condition est exécutée. Sinon, elle est ignorée. Ce masque recherche donc une séquence de lettre, éventuellement mis entre parenthèse.

Si la condition n'est pas une séquence de nombre, il faut que ce soit une assertion. Ce peut être une assertion positive ou négative, arrière ou avant.

Considérons le masque suivant (même conditions que le précédent) et avec deux alternatives en seconde ligne :

```
(?(?=[^a-z]*[a-z])
```

```
\d{2}[a-z]{3}-\d{2} | \d{2}-\d{2}-\d{2} )
```

La condition est une assertion avant positive, qui recherche une séquence optionnelle de caractère non-lettre. En d'autres termes, elle teste la présence d'au moins une lettre dans la chaîne sujet. Si une lettre est trouvée, la recherche se

poursuit avec la première alternative, et sinon, avec la seconde. Ce masque recherche des chaînes de la forme dd-aaa-dd ou dd-dd-dd, avec aaa qui sont des lettres, et dd qui sont des chiffres

COMMENTS

La séquence (?# marque le début des commentaires, qui se termine à la prochaine parenthèse fermante. Les parenthèses imbriquées ne sont pas autorisées. Les caractères entre ces délimiteurs ne jouent alors aucun rôle dans le masque.

Si l'option PCRE_EXTENDED est mise, les caractères dièses # non échappés en dehors d'une classe de caractères introduisent un commentaire qui continuera jusqu'à la prochaine ligne dans le masque.

PERFORMANCE

Certaines séquences de recherches sont plus efficaces que d'autres. Ainsi, il est plus efficace d'utiliser une classe de caractères telle que [aeiou] plutôt qu'une alternative (a|e|i|o|u).

En général, le masque le plus simple, qui permette la recherche désirée est généralement le plus efficace. Le livre de Jeffrey Friedl's contient de nombreuses études à propos de l'optimisation des expressions régulières.

Lorsqu'un masque commence par .* et que l'option PCRE_DOTALL est mise, le masque est implicitement ancré par PCRE, étant donné qu'il ne peut que rechercher au début de la chaîne. Cependant, si option PCRE_DOTALL n'est pas mise, PCRE ne peut faire aucune optimisation car le méta-caractère point (.). ne remplace pas une nouvelle ligne, et si la chaîne sujet contient des nouvelles lignes, le masque peut trouver une solution qui serait située juste après une de ces nouvelles lignes, et non pas seulement au début de la chaîne sujet. Par exemple, le masque, (.*).second

acceptera la chaîne "premier \net second" (avec \n qui remplace la nouvelle ligne), et la première chaîne capturée sera "et". Afin d'effectuer la recherche, PCRE va essayer d'appliquer le masque à partir de chaque début de ligne.

Si vous utilisez un tel masque avec des chaînes qui ne contiennent pas de caractères de nouvelles lignes, les meilleures performances seront atteintes avec l'option PCRE_DOTALL, ou en ancrant le masque avec ^.*. Cela évite à PCRE de scanner toute la chaîne pour rechercher un caractère de nouvelle ligne et recommencer la recherche.

10.52 Fonctions PDF

[\[Notes en ligne\]](#)

Vous disposez de fonctions PDF en PHP pour créer des fichiers PDF, pour peu que vous ayez la bibliothèque PDF de Thomas Merz (disponible à : <http://www.pdflib.com/pdflib/index.html> (site anglais)). Vous aurez aussi besoin des librairies [JPEG library](#); [the TIFF library](#), pour compiler cette librairie. Si vous utilisez pdflib 2.01, vérifiez l'installation de la librairie. Il ne devrait y avoir qu'un fichier (ou lien) : libPDF.so. La version 2.01 ne fait que créer une librairie appelée libpdf2.01.so, qui ne peut être trouvée lors du link du programme de test, lors de la configuration. Il vous faudra faire ce lien vous même, de libPDF.so vers libpdf2.01.so.

La version 2.20 de pdflib a introduit beaucoup de modifications dans ses API pour supporter les polices chinoises et japonaises. Cela modifie malheureusement les API du module PHP4 (mais pas le 3). Utilisez pdflib 2.20 qui gère correctement la mémoire lors de la génération de fichier PDF. Jusqu'à la publication de la version 3.0, pdflib peut être un peu instable. Le paramètre d'encodage de [pdf_set_font\(\)](#) a été changé en chaîne. Ainsi, à la place de '4', vous devrez utiliser 'winansi'.

Si vous utilisez pdflib 2.30 la fonction [pdf_set_text_matrix\(\)](#) a disparu. Elle n'est plus supportée. En général, c'est une bonne idée de consulter les notices de publications, pour connaître les modifications qui sont intervenues.

Reportez vous à l'excellente documentation de pdflib, disponible avec la distribution de pdflib ou à : <http://www.ifconnection.de/~tm/software/pdflib/PDFlib-0.6.PDF>. C'est une introduction très pratique des capacités de pdflib. La plus part des fonctions de pdflib se retrouvent dans PHP sous le même nom. De même, les paramètres sont identiques. Vous devez connaître les concepts de base de PDF ou de Postscript

pour utiliser efficacement ce module. Toutes les longueurs et coordonnées sont mesurées en points Postscript points. Il y a généralement 72 points PostScript par pouce, mais cela dépend en fait de la résolution d'affichage.

Il y a un autre module PHP pour créer des document PDF, basé sur la bibliothèque [FastIO's ClibPDF](#). Les API sont légèrement différentes. Reportez vous à la section [10.9 Fonctions ClibPDF](#) pour plus de détails. Actuellement, toutes les versions de la bibliothèque pdflib sont supportées. Il est préférable d'utiliser la nouvelle version, étant donné qu'elle comporte moins de bugs, et propose plus de fonctions que l'ancienne version. Malheureusement, les modifications dans les API de la bibliothèque ont été si importantes que certaines fonctions PHP ont dû être modifiées. Voici une petite liste des changements :

- La structure Info n'existe plus. Par conséquent, [pdf_get_info\(\)](#) est obsolète et les fonctions [pdf_set_info_creator\(\)](#), [pdf_set_info_title\(\)](#), [pdf_set_info_author\(\)](#), [pdf_set_info_subject\(\)](#) et [pdf_set_info_keywords\(\)](#) ne prennent plus de structure info comme premier paramètre, mais le document PDF. Cela signifie aussi que le document PDF doit être ouvert pour pouvoir appeler ces fonctions.
- La manière d'ouvrir un document a changé. La fonction [pdf_open\(\)](#) ne prend plus qu'un paramètre, qui est le pointeur de fichier, obtenu avec la fonction [fopen\(\)](#).

Il y a aussi quelques nouveautés avec la version 2.01 de pdflib qui devrait être couvertes par PHP. Quelques fonctions ne sont plus utilisées (e.g. [pdf_put_image\(\)](#)). Ne soyez pas surpris d'obtenir une alerte... Le module PDF introduit deux nouveaux types de variables (avec pdflib 2.x, seulement un nouveau type) C'est pdfdoc et pdfinfo (pdfinfo n'existe pas avec pdflib 2.x. pdfdoc est un pointeur de document PDF, et presque toutes les fonctions en ont besoin comme premier paramètre. pdfinfo contient des méta données à propos du document PDF document. Il doit exister avant d'appeler (*pdfinfo* n'existe pas si pdflib 2.x est utilisé. *pdfdoc* est un pointeur sur un document PDF et presque toutes les fonctions le requièrent comme premier paramètre. *pdfinfo* contient des méta-données sur les documents PDF. Elles doivent être fixées avant d'utiliser [pdf_open\(\)](#).

Note : *La suite n'est vraie qu'avec pdflib 0.6. Lisez le manuel de pdflib pour les nouveautés des nouvelles versions.*

Pour sauver des données dans un fichier PDF, vous devez fournir un fichier de type afm pour chaque police de caractère. Les fichiers Afm contiennent les définitions des polices Postscript. Par défaut, ces fichiers afm sont recherchés dans un dossier nommé 'fonts', près du dossier qui contient le script exécuté. (Encore une fois, ceci était vrai avec pdflib 0.6, mais ne le sera pas nécessairement avec les nouvelles versions).

La plupart des fonctions sont plutôt simples à utiliser. Le plus difficile est probablement la création d'un document PDF simple. Les exemples suivants vous aideront à repérer les premiers éléments. Ils utilisent les fonctions accessibles avec pdflib 0.6. Ils créent un exemple de test, test.pdf, d'une seule page. La page contient le texte "Times-Roman" en police renforcée, soulignée, de 30 pt.

Creation d'un document PDF avec pdflib 0.6

```
<?php
$fp = fopen("test.pdf", "w");
$info = PDF_get_info();
pdf_set_info_author($info, "Uwe Steinmann");
PDF_set_info_title($info, "Test for PHP wrapper of PDFlib 0.6");
PDF_set_info_author($info, "Name of Author");
pdf_set_info_creator($info, "See Author");
pdf_set_info_subject($info, "Testing");
$pdf = PDF_open($fp, $info);
PDF_begin_page($pdf, 595, 842);
PDF_add_outline($pdf, "Page 1");
pdf_set_font($pdf, "Times-Roman", 30, 4);
```

```
pdf_set_text_rendering($pdf, 1);
PDF_show_xy($pdf, "Times Roman outlined", 50, 750);
pdf_moveto($pdf, 50, 740);
pdf_lineto($pdf, 330, 740);
pdf_stroke($pdf);
PDF_end_page($pdf);
PDF_close($pdf);
fclose($fp);
echo "<A HREF=getPDF.php3>finished</A>";
?>
```

Le script `getPDF.php3` ne fait que produire un fichier PDF.

```
<?php
$fp = fopen("test.pdf", "r");
header("Content-type: application/pdf");
fpassthru($fp);
fclose($fp);
?>
```

La même chose avec `pdflib 2.x` ressemble à ce qui suit : *Création d'un document PDF avec `pdflib 2.x`*

```
<?php
$fp = fopen("test.pdf", "w");
$pdf = PDF_open($fp);
pdf_set_info_author($pdf, "Uwe Steinmann");
PDF_set_info_title($pdf, "Test for PHP wrapper of PDFlib 2.0");
PDF_set_info_author($pdf, "Name of Author");
pdf_set_info_creator($pdf, "See Author");
pdf_set_info_subject($pdf, "Testing");
PDF_begin_page($pdf, 595, 842);
PDF_add_outline($pdf, "Page 1");
pdf_set_font($pdf, "Times-Roman", 30, 4);
pdf_set_text_rendering($pdf, 1);
PDF_show_xy($pdf, "Times Roman outlined", 50, 750);
pdf_moveto($pdf, 50, 740);
pdf_lineto($pdf, 330, 740);
pdf_stroke($pdf);
PDF_end_page($pdf);
PDF_close($pdf);
fclose($fp);
echo "<A HREF=getPDF.php3>finished</A>";
?>
```

Ce script est identique à celui ci-dessus.

La distribution de `pdflib` contient d'autres exemples plus élaborés, qui crée des pages plus consistantes. Cet exemple convertit en PHP, et en utilisant `pdflib 2.x` ressemble à ce qui suit : (Vous pouvez retrouver cet exemple dans la documentation de [10.9 Fonctions ClibPDF: Exemple `pdfclock` de la distribution `pdflib 2.x`](#))

```
<?php
$pdffilename = "clock.pdf";
$radius = 200;
$margin = 20;
$pagecount = 40;
```

```

$fp = fopen($pdffilename, "w");
$pdf = pdf_open($fp);
pdf_set_info_creator($pdf, "pdf_clock.php3");
pdf_set_info_author($pdf, "Uwe Steinmann");
pdf_set_info_title($pdf, "Analog Clock");
while($pagecount-- > 0) {
pdf_begin_page($pdf, 2 * ($radius + $margin), 2 * ($radius + $margin));
pdf_set_transition($pdf, 4); /* wipe */
pdf_set_duration($pdf, 0.5);
pdf_translate($pdf, $radius + $margin, $radius + $margin);
pdf_save($pdf);
pdf_setrgbcolor($pdf, 0.0, 0.0, 1.0);
    /* minutes */
pdf_setlinewidth($pdf, 2.0);
for ($alpha = 0; $alpha < 360; $alpha += 6) {
pdf_rotate($pdf, 6.0);
pdf_moveto($pdf, $radius, 0.0);
pdf_lineto($pdf, $radius-$margin/3, 0.0);
pdf_stroke($pdf);
}
pdf_restore($pdf);
pdf_save($pdf);
    /* 5 minutes */
pdf_setlinewidth($pdf, 3.0);
for ($alpha = 0; $alpha < 360; $alpha += 30) {
pdf_rotate($pdf, 30.0);
pdf_moveto($pdf, $radius, 0.0);
pdf_lineto($pdf, $radius-$margin, 0.0);
pdf_stroke($pdf);
}
    $ltime = getdate();
    /* Dessine les heures */
pdf_save($pdf);
pdf_rotate($pdf, -((($ltime['minutes']/60.0)+$ltime['hours']-3.0)*30.0);
pdf_moveto($pdf, -$radius/10, -$radius/20);
pdf_lineto($pdf, $radius/2, 0.0);
pdf_lineto($pdf, -$radius/10, $radius/20);
pdf_closepath($pdf);
pdf_fill($pdf);
pdf_restore($pdf);
    /* draw minute hand */
pdf_save($pdf);
pdf_rotate($pdf, -((($ltime['seconds']/60.0)+$ltime['minutes']-15.0)*6.0);
pdf_moveto($pdf, -$radius/10, -$radius/20);
pdf_lineto($pdf, $radius * 0.8, 0.0);
pdf_lineto($pdf, -$radius/10, $radius/20);
pdf_closepath($pdf);
pdf_fill($pdf);
pdf_restore($pdf);
    /* draw second hand */
pdf_setrgbcolor($pdf, 1.0, 0.0, 0.0);
pdf_setlinewidth($pdf, 2);
pdf_save($pdf);
pdf_rotate($pdf, -((($ltime['seconds'] - 15.0) * 6.0));
pdf_moveto($pdf, -$radius/5, 0.0);
pdf_lineto($pdf, $radius, 0.0);
pdf_stroke($pdf);
pdf_restore($pdf);
    /* draw little circle at center */
pdf_circle($pdf, 0, 0, $radius/30);
pdf_fill($pdf);
pdf_restore($pdf);

```

```
pdf_end_page($pdf);
}
$pdf = pdf_close($pdf);
fclose($fp);
echo "<A HREF=getPDF.php3?filename=".$pdffilename.">finished</A>";
?>
```

Le script getPDF.php3 ne fait qu'afficher le fichier PDF.

```
<?php
$fp = fopen($filename, "r");
header("Content-type: application/pdf");
fpassthru($fp);
fclose($fp);
?>
```

10.52.1 PDF_get_info

[\[Notes en ligne\]](#) [\[Exemples\]](#)

info [pdf_get_info](#) (string *filename*)

[pdf_get_info\(\)](#) retourne une structure d'information de document PDF. Cette structure est la structure par défaut. Elle devra être remplie avec des informations adéquates, telle que l'auteur, le sujet etc.... Note : *Cette fonction n'est pas disponible si pdflib 2.x est activée.*

Voir aussi [pdf_set_info_creator\(\)](#), [pdf_set_info_author\(\)](#), [pdf_set_info_keywords\(\)](#), [pdf_set_info_title\(\)](#), [pdf_set_info_subject\(\)](#).

10.52.2 PDF_set_info_creator

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_info_creator](#) (info *info*, string *creator*)

[pdf_set_info_creator\(\)](#) affecte le champs titre de la structure info d'un document PDF. Cette fonction doit être appelée après [pdf_get_info\(\)](#) et avant [pdf_open\(\)](#). L'appeler après [pdf_open\(\)](#) sera sans effet sur le document.

Note : *Cette fonction ne fait pas partie de PDF library.*

Note : *Cette fonction prend un autre type de premier paramètre si pdflib 2.x est activé. Le premier paramètre doit alors être un identifiant de document PDF document, retourné par [pdf_open\(\)](#). Par conséquent, [pdf_open\(\)](#) doit avoir été appelé avant cette fonction.*

Voir aussi [pdf_get_info\(\)](#), [pdf_set_info_keywords\(\)](#), [pdf_set_info_title\(\)](#), [pdf_set_info_subject\(\)](#).

10.52.3 PDF_set_info_title

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_info_title](#) (info *info*, string *title*)

[pdf_set_info_title\(\)](#) modifie le sujet d'un document PDF. Cette fonction doit être appelée après [pdf_get_info\(\)](#) mais avant [pdf_open\(\)](#). Si vous l'appellez après [pdf_open\(\)](#), elle n'aura pas d'effet sur le document. Note : *Cette fonction ne fait pas partie de PDF library.*

Note : *Cette fonction prend un autre type de premier paramètre si pdflib 2.x est activé. Le premier paramètre doit alors être un identifiant de document PDF document, retourné par [pdf_open\(\)](#). Par conséquent, [pdf_open\(\)](#) doit avoir été appelé avant cette fonction.*

Voir aussi [pdf_get_info\(\)](#), [pdf_set_info_creator\(\)](#), [pdf_set_info_author\(\)](#), [pdf_set_info_keywords\(\)](#), [pdf_set_info_subject\(\)](#).

10.52.4 PDF_set_info_subject

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_info_subject](#)(info *info*, string *subject*)

[pdf_set_info_subject\(\)](#) modifie le sujet d'un document PDF. Cette fonction doit être appelée après [pdf_get_info\(\)](#) et avant [pdf_open\(\)](#), sinon, elle sera sans effet. Note : *Cette fonction ne fait pas partie de PDF library.*

Note : *Cette fonction prend un autre type de premier paramètre si pdflib 2.x est activé. Le premier paramètre doit alors être un identifiant de document PDF document, retourné par [pdf_open\(\)](#). Par conséquent, [pdf_open\(\)](#) doit avoir été appelé avant cette fonction.*

Voir aussi [pdf_get_info\(\)](#), [pdf_set_info_creator\(\)](#), [pdf_set_info_author\(\)](#), [pdf_set_info_title\(\)](#), [pdf_set_info_keywords\(\)](#).

10.52.5 PDF_set_info_keywords

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_info_keywords](#)(info *info*, string *keywords*)

[pdf_set_info_keywords\(\)](#) affecte le champs mots-clé de la structure d'un document PDF. Cette fonction doit être appelée après [pdf_get_info\(\)](#) et avant [pdf_open\(\)](#), sinon, elle sera sans effet. Note : *Cette fonction ne fait pas partie de PDF library.*

Note : *Cette fonction prend un autre type de premier paramètre si pdflib 2.x est activé. Le premier paramètre doit alors être un identifiant de document PDF document, retourné par [pdf_open\(\)](#). Par conséquent, [pdf_open\(\)](#) doit avoir été appelé avant cette fonction.*

Voir aussi [pdf_get_info\(\)](#), [pdf_set_info_creator\(\)](#), [pdf_set_info_author\(\)](#), [pdf_set_info_title\(\)](#), [pdf_set_info_subject\(\)](#).

10.52.6 PDF_set_info_author

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_info_author](#)(info *info*, string *author*)

[pdf_set_info_author\(\)](#) affecte le champs auteur de la structure d'un document PDF. Cette fonction doit être appelée après [pdf_get_info\(\)](#) et avant [pdf_open\(\)](#), sinon, elle sera sans effet. Note : *Cette fonction ne fait pas partie de PDF library.*

Note : Cette fonction prend un autre type de premier paramètre si *pdflib 2.x* est activé. Le premier paramètre doit alors être un identifiant de document PDF *document*, retourné par [pdf_open\(\)](#). Par conséquent, [pdf_open\(\)](#) doit avoir été appelé avant cette fonction.

Voir aussi [pdf_get_info\(\)](#), [pdf_set_info_creator\(\)](#), [pdf_set_info_keywords\(\)](#), [pdf_set_info_title\(\)](#), [pdf_set_info_subject\(\)](#).

10.52.7 PDF open

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pdf_open](#)(int *file*, int *info*)

[pdf_open\(\)](#) ouvre un nouveau document PDF. Le fichier correspondant doit avoir été ouvert avec [fopen\(\)](#) et le pointeur de fichier est passé en argument *file*. *info* est une structure info créée par [pdf_get_info\(\)](#). Cette structure info sera effacée à partir de cette fonction. Note : Cette fonction ne fait pas partie de PDF library.

Note : Cette fonction prend un autre type de premier paramètre si *pdflib 2.x* est activé. Le premier paramètre doit alors être un identifiant de document PDF *document*, retourné par [pdf_open\(\)](#). Par conséquent, [pdf_open\(\)](#) doit avoir été appelé avant cette fonction.

Voir aussi [fopen\(\)](#), [pdf_get_info\(\)](#), [pdf_close\(\)](#).

10.52.8 PDF close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_close](#)(int *pdf document*)

[pdf_close\(\)](#) ferme un document PDF. Note : A cause de la programmation buggée de *pdflib 0.6* fermer un document PDF ferme aussi le fichier associé. Cela ne devrait pas être le cas, puisque *pdflib* n'a pas ouvert le fichier, mais s'attend à un fichier déjà ouvert, lorsque [pdf_open\(\)](#) est appelée. Par conséquent, il ne devrait pas fermer le fichier. Afin de résoudre de problème, mettez en commentaire la ligne 190 dans le fichier *p_basic.c* de *pdflib 0.6* jusqu'à ce qu'une nouvelle version corrige ce bug.

Note : Cette fonction n'a pas besoin de patch avec *pdflib*, si *pdflib 2.0* est activée.

Voir aussi [pdf_open\(\)](#), [fclose\(\)](#).

10.52.9 PDF begin page

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_begin_page](#)(int *pdf document*, double *height*, double *width*)

[pdf_begin_page\(\)](#) commence une nouvelle page avec la taille *height* et la largeur *width*.

Voir aussi [pdf_end_page\(\)](#).

10.52.10 PDF end page

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_end_page](#)(int *pdf document*)

[pdf_end_page\(\)](#) termine une page. Une fois qu'une page a été fermée, elle ne peut pas être modifiée. Voir aussi [pdf_begin_page\(\)](#).

10.52.11 PDF show

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_show\(\)](#) (int *pdf document*, string *text*)

[pdf_show\(\)](#) affiche le texte *text* avec la position courante, et avec la police courante. Voir aussi [pdf_show_xy\(\)](#), [pdf_set_text_pos\(\)](#), [pdf_set_font\(\)](#).

10.52.12 PDF show boxed

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_show_boxed\(\)](#) (int *pdf document*, string *text*, double *x-coor*, double *y-coor*, double *width*, double *height*, string *mode*)

[pdf_show_boxed\(\)](#) affiche le texte *text* dans un rectangle, dont le coin inférieur gauche est aux coordonnées (*x-coor*, *y-coor*). Les dimensions du rectangle sont *height* et *width*. Le paramètre *mode* indique le type de *text*. Si *width* et *height* sont à zéro, le mode *mode* peut être "left" (gauche), "right" (droite) ou "center" (centré). *width* ou *height* sont différents pouvant prendre les valeurs de "justify" (justification) ou "fulljustify" (justification complète).

Voir aussi [pdf_show\(\)](#), [pdf_show_xy\(\)](#).

10.52.13 PDF show xy

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_show_xy\(\)](#) (int *pdf document*, string *text*, double *x-coor*, double *y-coor*)

[pdf_show_xy\(\)](#) affiche le texte *text* à la position donnée par les coordonnées (*x-coor*, *y-coor*). Voir aussi [pdf_show\(\)](#).

10.52.14 PDF set font

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_font\(\)](#) (int *pdf document*, string *font name*, double *size*, string *encoding*, int *embed*)

[pdf_set_font\(\)](#) sélectionne la police, sa taille et son encodage. Il vous faudra fournir des fichiers Adobe Font Metrics (afm) comme police, dans le dossier de polices (par défaut ./fonts). Si vous utilisez pdflib 0.6, vous devrez fournir des fichiers Adobe Font Métrique (afm-files) pour les polices, dans le chemin de police (par défaut, ./fonts). Si vous utilisez php versin 3 ou une version plus ancienne que la version 2.20 de pdflib, le quatrième paramètre *encoding* peut prendre les valeurs suivantes : 0 = builtin, 1 = pdfdoc, 2 = macroman, 3 = macexpert, 4 = winansi. Un encodage plus grand que 4 et inférieur à 0 sera transformé en 'winansi'. 'winansi' est souvent un bon choix. Si vous utilisez PHP version 4 et une version plus ancienne que la version 2.20 de pdflib le quatrième paramètre *encoding* est une chaîne : 'builtin', 'pdfdoc', 'macroman', 'macexpert', 'winansi'. Si le dernier paramètre est à 1, la police est intégrée dans le document. Sinon, elle ne le sera pas. Incorporer

une police dans un document est une bonne idée si la police n'est pas répandue, ou si vous ne pouvez pas vous assurer que la personne qui regardera votre document peut accéder à cette police.

Note : Cette fonction doit être appelée après [pdf_begin_page\(\)](#) pour créer un document PDF valide.

[10.52.15 PDF_set_leading](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_leading](#)(int *pdf document*, double *distance*)

[pdf_set_leading\(\)](#) permet de choisir la distance entre les lignes du texte. Cette valeur sera utilisée si du texte est affiché par [pdf_continue_text\(\)](#).

Voir aussi [pdf_continue_text\(\)](#).

[10.52.16 PDF_set_parameter](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_parameter](#)(int *pdf document*, string *name*, string *value*)

[pdf_set_parameter\(\)](#) fixe certaines valeurs de pdglib.

[10.52.17 PDF_set_text_rendering](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_text_rendering](#)(int *pdf document*, int *mode*)

[pdf_set_text_rendering\(\)](#) détermine le rendu du texte. Les valeurs possibles pour *mode* sont 0=fill text (texte plein), 1=stroke text (???), 2=fill and stroke text (texte plein et stroke), 3=invisible, 4=texte plein, et ajouté au chemin, 5=stroke text, ajouté au chemin, 6=texte plein et stroke, ajouté au chemin, 7=ajouté au chemin.

[10.52.18 PDF_set_horiz_scaling](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_horiz_scaling](#)(int *pdf document*, double *scale*)

[pdf_set_horiz_scaling\(\)](#) fixe l'échelle horizontale du texte, à *scale* en pourcentage.

[10.52.19 PDF_set_text_rise](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_text_rise](#)(int *pdf document*, double *rise*)

[pdf_set_text_rise\(\)](#) fixe l'élévation du texte à *rise* points.

[10.52.20 PDF_set_text_matrix](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_text_matrix](#)(int *pdf document*, array *matrix*)

[pdf_set_text_matrix\(\)](#) choisit la matrice de texte : la matrice de texte détermine les transformations de la police courante. La matrice est un tableau de 6 éléments.

10.52.21 PDF_set_text_pos

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_text_pos](#)(int *pdf document*, double *x-coor*, double *y-coor*)

[pdf_set_text_pos\(\)](#) choisit la position du texte qui sera utilisée lors du prochain [pdf_show\(\)](#). Voir aussi [pdf_show\(\)](#), [pdf_show_xy\(\)](#).

10.52.22 PDF_set_char_spacing

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_char_spacing](#)(int *pdf document*, double *space*)

[pdf_set_char_spacing\(\)](#) fixe l'espacement des caractères. Voir aussi [pdf_set_word_spacing\(\)](#), [pdf_set_leading\(\)](#).

10.52.23 PDF_set_word_spacing

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_word_spacing](#)(int *pdf document*, double *space*)

[pdf_set_word_spacing\(\)](#) fixe l'espacement des mots. Voir aussi [pdf_set_char_spacing\(\)](#), [pdf_set_leading\(\)](#).

10.52.24 PDF_skew

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_skew](#)(int *pdf document*, double *alpha*, double *beta*)

[pdf_skew\(\)](#) modifie le système de coordonnées, en faisant une rotation d'angle *alpha* pour les (x) et d'angle *beta* pour les (y), en degrés. *alpha* et *beta* ne peuvent pas prendre les valeurs de 90 ou 270 degrés.

10.52.25 PDF_continue_text

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_continue_text](#)(int *pdf document*, string *text*)

[pdf_continue_text\(\)](#) affiche le texte *text* sur une nouvelle ligne. La distance entre les lignes peut être choisie avec [pdf_set_leading\(\)](#). Voir aussi [pdf_show_xy\(\)](#), [pdf_set_leading\(\)](#), [pdf_set_text_pos\(\)](#).

10.52.26 PDF_stringwidth

[\[Notes en ligne\]](#) [\[Exemples\]](#)

double [pdf_stringwidth](#)(int *pdf document*, string *text*)

[pdf_stringwidth\(\)](#) retourne la largeur du texte *text* avec la police courante. Il faut qu'une police ait été choisie auparavant.

Voir aussi [pdf_set_font\(\)](#).

10.52.27 PDF_save

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_save](#)(int *pdf document*)

[pdf_save\(\)](#) enregistre l'environnement courant. Le fonctionnement est identique à la commande postscript gsave. Très pratique si vous voulez faire une translation ou une rotation d'un objet, sans affecter les autres.

[pdf_save\(\)](#) sera toujours suivi d'un [pdf_restore\(\)](#).

Voir aussi [pdf_restore\(\)](#).

10.52.28 PDF_restore

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_restore](#)(int *pdf document*)

[pdf_restore\(\)](#) restaure un environnement sauvé par [pdf_save\(\)](#). Cela fonctionne de manière identique à la commande postscript grestore. Très pratique lorsque vous vous faire des translations ou des rotations sans affecter les autres objets. *Sauver et Restaurer un environnement*

```
<?php PDF_save($pdf);
// tout un lot de rotations, translations, transformations...
PDF_restore($pdf) ?>
```

Voir aussi [pdf_save\(\)](#).

10.52.29 PDF_translate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_translate](#)(int *pdf document*, double *x-coor*, double *y-coor*)

[pdf_translate\(\)](#) place l'origine du système de coordonnées au point (*x-coor*, *y-coor*). L'exemple suivant trace une ligne de (0, 0) à (200, 200) par rapport aux coordonnées initiales. Il faut aussi désigner le point courant après [pdf_translate\(\)](#) et avant de commencer à dessiner les objets. *Translation*

```
<?php PDF_moveto($pdf, 0, 0);
```

```
PDF_lineto($pdf, 100, 100);
PDF_stroke($pdf);
PDF_translate($pdf, 100, 100);
PDF_moveto($pdf, 0, 0);
PDF_lineto($pdf, 100, 100);
PDF_stroke($pdf);
?>
```

10.52.30 PDF_scale

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_scale](#)(int *pdf document*, double *x-scale*, double *y-scale*)

[pdf_scale\(\)](#) choisi l'échelle dans les deux directions. L'exemple suivant multiplie l'échelle par 72. La ligne suivante sera dessinée sur un pouce (2.54 cm) de large. *Mise à l'échelle*

```
<?php PDF_scale($pdf, 72.0, 72.0);
PDF_lineto($pdf, 1, 1);
PDF_stroke($pdf);
?>
```

10.52.31 PDF_rotate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_rotate](#)(int *pdf document*, double *angle*)

[pdf_rotate\(\)](#) choisi la rotation en degré.

10.52.32 PDF_setflat

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setflat](#)(int *pdf document*, double *value*)

[pdf_setflat\(\)](#) choisi la platitude, et lui affecte une valeur entre 0 et 100.

10.52.33 PDF_setlinejoin

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setlinejoin](#)(int *pdf document*, long *value*)

[pdf_setlinejoin\(\)](#) choisi le paramètre "linejoin", et lui affecte une valeur entre 0 et 2.

[10.52.34 PDF_setlinecap](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setlinecap](#) (int *pdf document*, int *value*)

[pdf_setlinecap\(\)](#) affecte au paramètre *linecap* une valeur entre 0 et 2.

[10.52.35 PDF_setmiterlimit](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setmiterlimit](#) (int *pdf document*, double *value*)

[pdf_setmiterlimit\(\)](#) choisi la "miter limit" et lui affecte une valeur supérieure à 1.

[10.52.36 PDF_setlinewidth](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setlinewidth](#) (int *pdf document*, double *width*)

[pdf_setlinewidth\(\)](#) affecte à largeur de ligne la valeur *width*.

[10.52.37 PDF_setdash](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setdash](#) (int *pdf document*, double *white*, double *black*)

[pdf_setdash\(\)](#) choisi les caractères de remplissage, et affecte *white* comme caractère invisible, et *black* comme caractère de remplissage. Si les deux sont à zéros, une ligne continue est affichée.

[10.52.38 PDF_moveto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_moveto](#) (int *pdf document*, double *x-coor*, double *y-coor*)

[pdf_moveto\(\)](#) déplace le point courant à la position (*x-coor*, *y-coor*).

[10.52.39 PDF_curveto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_curveto](#) (int *pdf document*, double *x1*, double *y1*, double *x2*, double *y2*, double *x3*, double *y3*)

[pdf_curveto\(\)](#) dessine une courbe de Bezier entre le point courant et le point (*x3*, *y3*) en utilisant les points de contrôle (*x1*, *y1*) et (*x2*, *y2*).

Voir aussi [pdf_moveto\(\)](#), [pdf_lineto\(\)](#), [pdf_stroke\(\)](#).

10.52.40 PDF `lineto`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_lineto](#) (int *pdf document*, double *x-coor*, double *y-coor*)

[pdf_lineto\(\)](#) dessine une ligne entre le point courant et le point de coordonnées (*x-coor*, *y-coor*).
Voir aussi [pdf_moveto\(\)](#), [pdf_curveto\(\)](#), [pdf_stroke\(\)](#).

10.52.41 PDF `circle`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_circle](#) (int *pdf document*, double *x-coor*, double *y-coor*, double *radius*)

[pdf_circle\(\)](#) dessine un cercle de centre (*x-coor*, *y-coor*), et de rayon *radius*.
Voir aussi [pdf_arc\(\)](#), [pdf_stroke\(\)](#).

10.52.42 PDF `arc`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_arc](#) (int *pdf document*, double *x-coor*, double *y-coor*, double *radius*, double *start*, double *end*)

[pdf_arc\(\)](#) dessine un arc de cercle, de centre (*x-coor*, *y-coor*) et de rayon *radius*, en commençant à l'angle *start* et finissant à l'angle *end*.
Voir aussi [pdf_circle\(\)](#), [pdf_stroke\(\)](#).

10.52.43 PDF `rect`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_rect](#) (int *pdf document*, double *x-coor*, double *y-coor*, double *width*, double *height*)

[pdf_rect\(\)](#) dessine un rectangle un rectangle de coin inférieur gauche de coordonnées (*x-coor*, *y-coor*). Sa longueur vaut *width*. Et sa largeur *height*.
Voir aussi [pdf_stroke\(\)](#).

10.52.44 PDF `closepath`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_closepath](#) (int *pdf document*)

[pdf_closepath\(\)](#) ferme et clos le chemin courant. Cela signifie qu'une ligne va être ajoutée entre le point courant et le premier point du chemin. De nombreuses fonctions telles que [pdf_moveto\(\)](#), [pdf_circle\(\)](#) et [pdf_rect\(\)](#) démarre un nouveau chemin.

10.52.45 PDF `stroke`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_stroke](#)(int *pdf document*)

[pdf_stroke\(\)](#) dessine une ligne le long du chemin. Le chemin courant est la somme de toutes les lignes dessinées. Sans cette fonction, la ligne de chemin ne sera pas dessinée.

Voir aussi [pdf_closepath\(\)](#), [pdf_closepath_stroke\(\)](#).

10.52.46 PDF `closepath_stroke`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_closepath_stroke](#)(int *pdf document*)

[pdf_closepath_stroke\(\)](#) est une combinaison de [pdf_closepath\(\)](#) et [pdf_stroke\(\)](#). Elle ferme aussi le chemin.

Voir aussi [pdf_closepath\(\)](#), [pdf_stroke\(\)](#).

10.52.47 PDF `fill`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_fill](#)(int *pdf document*)

[pdf_fill\(\)](#) remplit l'intérieur du chemin courant avec la couleur courante.

Voir aussi [pdf_closepath\(\)](#), [pdf_stroke\(\)](#), [pdf_setgray_fill\(\)](#), [pdf_setgray\(\)](#), [pdf_setrgbcolor_fill\(\)](#), [pdf_setrgbcolor\(\)](#).

10.52.48 PDF `fill_stroke`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_fill_stroke](#)(int *pdf document*)

[pdf_fill_stroke\(\)](#) remplit l'intérieur du chemin courant avec la couleur courante, puis dessine le chemin courant.

Voir aussi [pdf_closepath\(\)](#), [pdf_stroke\(\)](#), [pdf_fill\(\)](#), [pdf_setgray_fill\(\)](#), [pdf_setgray\(\)](#), [pdf_setrgbcolor_fill\(\)](#), [pdf_setrgbcolor\(\)](#).

10.52.49 PDF `closepath_fill_stroke`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_closepath_fill_stroke](#)(int *pdf document*)

[pdf_closepath_fill_stroke\(\)](#) clos le chemin, le remplit avec la couleur courante, et dessine le chemin.

Voir aussi [pdf_closepath\(\)](#), [pdf_stroke\(\)](#), [pdf_fill\(\)](#), [pdf_setgray_fill\(\)](#), [pdf_setgray\(\)](#), [pdf_setrgbcolor_fill\(\)](#), [pdf_setrgbcolor\(\)](#).

[10.52.50 PDF_endpath](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_endpath](#) (int *pdf document*)

[pdf_endpath\(\)](#) ferme le chemin courant mais ne le clos pas.
Voir aussi [pdf_closepath\(\)](#).

[10.52.51 PDF_clip](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_clip](#) (int *pdf document*)

[pdf_clip\(\)](#) aligne tous les dessins sur le chemin courant.

[10.52.52 PDF_setgray_fill](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setgray_fill](#) (int *pdf document*, double *gray value*)

[pdf_setgray_fill\(\)](#) choisi la couleur grise comme couleur de remplissage.
Voir aussi [pdf_setrgbcolor_fill\(\)](#).

[10.52.53 PDF_setgray_stroke](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setgray_stroke](#) (int *pdf document*, double *gray value*)

[pdf_setgray_stroke\(\)](#) Fixe la couleur de dessin à un niveau de gris.
Voir aussi [pdf_setrgbcolor_stroke\(\)](#).

[10.52.54 PDF_setgray](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setgray](#) (int *pdf document*, double *gray value*)

[pdf_setgray\(\)](#) choisi la couleur grise comme couleur de remplissage et de dessin.
Voir aussi [pdf_setrgbcolor_stroke\(\)](#), [pdf_setrgbcolor_fill\(\)](#).

[10.52.55 PDF_setrgbcolor_fill](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setrgbcolor_fill](#) (int *pdf document*, double *red value*, double *green value*, double *blue value*)

[pdf_setrgbcolor_fill\(\)](#) choisi la couleur rgb comme couleur de remplissage.

Voir aussi [pdf_setrgbcolor_fill\(\)](#).

[10.52.56 PDF_setrgbcolor_stroke](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setrgbcolor_stroke](#) (int *pdf document*, double *red value*, double *green value*, double *blue value*)

[pdf_setrgbcolor_stroke\(\)](#) choisi la couleur rgb comme couleur de remplissage.

Voir aussi [pdf_setrgbcolor_stroke\(\)](#).

[10.52.57 PDF_setrgbcolor](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setrgbcolor](#) (int *pdf document*, double *red value*, double *green value*, double *blue value*)

[pdf_setrgbcolor_stroke\(\)](#) choisi la couleur rgb comme couleur de remplissage.

Voir aussi [pdf_setrgbcolor_stroke\(\)](#), [pdf_setrgbcolor_fill\(\)](#).

[10.52.58 PDF_add_outline](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pdf_add_outline](#) (int *pdf document*, string *text*, int *parent*, int *open*)

[pdf_add_outline\(\)](#) ajoute un signet de nom *text* sur la page courante, et au point courant. Le signet est inséré comme un fils de la page *parent* et est ouvert par défaut si *open* n'est pas 0. La valeur retournée est un identifiant du signet, qui pourra être réutilisé comme parent d'autres signets. Vous pouvez ainsi créer des hiérarchies de signets.

Malheureusement, pdf-lib ne copie pas la chaîne, ce qui force PHP à allouer la mémoire. Actuellement, cette mémoire n'est jamais libérée par une fonction PDF, mais prise en charge par le gestionnaire PHP.

[10.52.59 PDF_set_transition](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_transition](#) (int *pdf document*, int *transition*)

[pdf_set_transition\(\)](#) Fixe le mode de transition entre les pages. La valeur de *transition* peut être :

- 0 : aucune transition,
- 1 : deux lignes en travers de l'écran représente la page,
- 2 : plusieurs lignes en travers de l'écran représente la page,
- 3 : une boîte représente la page,
- 4 : une seule ligne en travers de l'écran représente la page,
- 5 : l'ancienne page se dissout pour révéler la nouvelle,
- 6 : l'effet page d'un coin à l'autre
- 7 : l'ancienne page est simplement remplacée par la nouvelle (valeur par défaut)

Voir aussi [pdf_set_duration\(\)](#).

[10.52.60 PDF_set_duration](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_duration](#)(int *pdf document*, double *duration*)

[pdf_set_duration\(\)](#) choisi la durée de transition, en secondes, entre deux pages.

Voir aussi [pdf_set_transition\(\)](#).

[10.52.61 PDF_open_gif](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pdf_open_gif](#)(int *pdf document*, string *filename*)

[pdf_open_gif\(\)](#) ouvre et charge l'image GIF du fichier *filename*. Le format doit être GIF. La fonction retourne un identifiant d'image PDF : ***Inclusion d'un image GIF***

```
<?php
$im = PDF_open_gif($pdf, "test.gif");
pdf_place_image($pdf, $im, 100, 100, 1);
pdf_close_image($pdf, $im);
?>
```

Voir aussi [pdf_close_image\(\)](#), [pdf_open_jpeg\(\)](#), [pdf_open_memory_image\(\)](#), [pdf_execute_image\(\)](#), [pdf_place_image\(\)](#), [pdf_put_image\(\)](#).

[10.52.62 PDF_open_memory_image](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pdf_open_memory_image](#)(int *pdf document*, int *image*)

[pdf_open_memory_image\(\)](#) prend comme argument une image créée avec les fonctions PHP, et la rend disponible pour le document PDF. La fonction retourne un identifiant PDF d'image. ***Inclusion d'une image mémoire***

```
<?php
$im = ImageCreate(100, 100);
$col = ImageColorAllocate($im, 80, 45, 190);
ImageFill($im, 10, 10, $col);
$pim = PDF_open_memory_image($pdf, $im);
ImageDestroy($im);
pdf_place_image($pdf, $pim, 100, 100, 1);
pdf_close_image($pdf, $pim);
?>
```

Voir aussi [pdf_close_image\(\)](#), [pdf_open_jpeg\(\)](#), [pdf_open_gif\(\)](#), [pdf_execute_image\(\)](#), [pdf_place_image\(\)](#), [pdf_put_image\(\)](#).

10.52.63 PDF_open_jpeg

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pdf_open_jpeg](#)(int *pdf document*, string *filename*)

[pdf_open_jpeg\(\)](#) ouvre et charge l'image JPEG du fichier *filename*. Le format de l'image doit être JPEG. La fonction retourne un identifiant d'image PDF.

Voir aussi [pdf_close_image\(\)](#), [pdf_open_gif\(\)](#), [pdf_open_memory_image\(\)](#), [pdf_execute_image\(\)](#), [pdf_place_image\(\)](#), [pdf_put_image\(\)](#).

10.52.64 PDF_close_image

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_close_image](#)(int *image*)

[pdf_close_image\(\)](#) ferme une image qui a été ouverte par [pdf_open_gif\(\)](#) ou [pdf_open_jpeg\(\)](#).

Voir aussi [pdf_open_jpeg\(\)](#), [pdf_open_gif\(\)](#), [pdf_open_memory_image\(\)](#).

10.52.65 PDF_place_image

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_place_image](#)(int *pdf document*, int *image*, double *x-coor*, double *y-coor*, double *scale*)

[pdf_place_image\(\)](#) place l'image *image* dans la page courante, à la position (*x-coor*, *x-coor*). L'image peut changer d'échelle simultanément.

Voir aussi [pdf_put_image\(\)](#).

10.52.66 PDF_put_image

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_put_image](#)(int *pdf document*, int *image*)

[pdf_put_image\(\)](#) enregistre une image dans un fichier PDF pour utilisation ultérieure. L'image n'est pas visible. On peut l'afficher avec la fonction [pdf_execute_image\(\)](#), et aussi souvent que désiré. Cela permet d'utiliser plusieurs fois la même image, sans augmenter la taille du fichier. [pdf_put_image\(\)](#) et [pdf_execute_image\(\)](#) est fortement recommandé pour les images de grande taille (plusieurs ko) si elles sont affichées plus d'une fois dans le document. Note : *Cette fonction n'a plus de sens avec la version 2.01 de pdflib. Elle ne fera que retourner une alerte.*

Voir aussi [pdf_put_image\(\)](#), [pdf_place_image\(\)](#), [pdf_execute_image\(\)](#).

10.52.67 PDF execute image

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_execute_image](#)(int *pdf document*, int *image*, double *x-coor*, double *y-coor*, double *scale*)

[pdf_execute_image\(\)](#) affiche une image qui a été enregistrée dans le document PDF, avec la fonction [pdf_put_image\(\)](#). L'image est implantée dans la page courante, et aux coordonnées données.

L'image peut être changée d'échelle en même temps qu'elle est affichée. Une échelle de 1.0 affichera l'image à sa taille d'origine. Note : *Cette fonction n'a plus de sens avec la version 2.01 de pdflib. Elle ne fera que retourner une alerte.*

Affichage multiple d'une image.

```
<?php
$im = ImageCreate(100, 100);
$coll = ImageColorAllocate($im, 80, 45, 190);
ImageFill($im, 10, 10, $coll);
$pim = PDF_open_memory_image($pdf, $im);
pdf_put_image($pdf, $pim);
pdf_execute_image($pdf, $pim, 100, 100, 1);
pdf_execute_image($pdf, $pim, 200, 200, 2);
pdf_close_image($pdf, $pim);
?>
```

10.52.68 pdf add annotation

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_add_annotation](#)(int *pdf document*, double *llx*, double *lly*, double *urx*, double *ury*, string *title*, string *content*)

[pdf_add_annotation\(\)](#) ajoute une note, dont le coin inférieur droit est (*llx*, *lly*) et le coin supérieur droit est (*urx*, *ury*).

10.53 Verisign Payflow Pro functions

[\[Notes en ligne\]](#)

This extension allows you to process credit cards and other financial transactions using Verisign Payment Services, formerly known as Signio (<http://www.verisign.com/payment/>).

These functions are only available if PHP has been compiled with the `--with-pfpro[=DIR]` option. You will require the appropriate SDK for your platform, which may be downloaded [from within the manager interface](#) once you have registered.

Once you have downloaded the SDK you should copy the files from the ``lib'` directory of the distribution. Copy the header file ``pfpro.h'` to ``/usr/local/include'` and the library file ``libpfpro.so'` to ``/usr/local/lib'`.

When using these functions, you may omit calls to [pfpro_init\(\)](#) and [pfpro_cleanup\(\)](#) as this extension will do so automatically if required. However the functions are still available in case you are processing a number of

transactions and require fine control over the library. You may perform any number of transactions using [pfpro_process\(\)](#) between the two.

These functions have been added in PHP 4.0.2.

Note : *These functions only provide a link to Verisign Payment Services. Be sure to read the Payflow Pro Developers Guide for full details of the required parameters.*

[10.53.1 pfpro_init](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pfpro_init](#)

[pfpro_init\(\)](#) is used to initialise the Payflow Pro library. You may omit this call, in which case this extension will automatically call [pfpro_init\(\)](#) before the first transaction.

See also [pfpro_cleanup\(\)](#).

[10.53.2 pfpro_cleanup](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pfpro_cleanup](#)

[pfpro_cleanup\(\)](#) is used to shutdown the Payflow Pro library cleanly. It should be called after you have processed any transactions and before the end of your script. However you may omit this call, in which case this extension will automatically call [pfpro_cleanup\(\)](#) after your script terminates.

See also [pfpro_init\(\)](#).

[10.53.3 pfpro_process](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [pfpro_process](#) (array *parameters*, string *address*, int *port*, int *timeout*, string *proxy address*, int *proxy port*, string *proxy logon*, string *proxy password*)

Returns: An associative array containing the response

[pfpro_process\(\)](#) processes a transaction with Payflow Pro. The first parameter is an associative array containing keys and values that will be encoded and passed to the processor.

The second parameter is optional and specifies the host to connect to. By default this is "test.signio.com", so you will certainly want to change this to "connect.signio.com" in order to process live transactions.

The third parameter specifies the port to connect on. It defaults to 443, the standard SSL port.

The fourth parameter specifies the timeout to be used, in seconds. This defaults to 30 seconds. Note that this timeout appears to only begin once a link to the processor has been established and so your script could potentially continue for a very long time in the event of DNS or network problems.

The fifth parameter, if required, specifies the hostname of your SSL proxy. The sixth parameter specifies the port to use.

The seventh and eighth parameters specify the logon identity and password to use on the proxy.

The function returns an associative array of the keys and values in the response.

Note : *Be sure to read the Payflow Pro Developers Guide for full details of the required parameters.*

Payflow Pro example

```

<?php
pfpro_init();
$transaction = array(USER      => 'mylogin',
PWD       => 'mypassword',
TRXTYPE  => 'S',
TENDER   => 'C',
AMT      => 1.50,
ACCT     => '4111111111111111',
EXPDATE  => '0904'
);

$response = pfpro_process($transaction);
if (!$response) {
die("Couldn't establish link to Verisign.\n");
}
echo "Verisign response code was ".$response[RESULT];
echo ", which means: ".$response[RESPMSG]."\n";
echo "\nThe transaction request: ";
print_r($transaction);
echo "\nThe response: ";
print_r($response);
pfpro_cleanup();
?>

```

[10.53.4 pfpro_process_raw](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pfpro_process_raw](#) (string *parameters*, string *address*, int *port*, int *timeout*, string *proxy address*, int *proxy port*, string *proxy logon*, string *proxy password*)

Returns: A string containing the response.

[pfpro_process_raw\(\)](#) processes a raw transaction string with Payflow Pro. You should really use [pfpro_process\(\)](#) instead, as the encoding rules of these transactions are non-standard.

The first parameter in this case is a string containing the raw transaction request. All other parameters are the same as with [pfpro_process\(\)](#). The return value is a string containing the raw response.

Note : *Be sure to read the Payflow Pro Developers Guide for full details of the required parameters and encoding rules. You would be well advised to use [pfpro_process\(\)](#) instead.*

Payflow Pro raw example

```

<?php
pfpro_init();
$response = pfpro_process("USER=mylogin&PWD[5]=m&ndy&TRXTYPE=S&TENDER=C&AMT=1.50&ACCT=4111111111111111");
if (!$response) {
die("Couldn't establish link to Verisign.\n");
}
echo "Verisign raw response was ".$response;
pfpro_cleanup();
?>

```

[10.53.5 pfpro_version](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pfpro_version](#)

[pfpro_version\(\)](#) returns the version string of the Payflow Pro library. At the time of writing, this was L211.

10.54 Fonctions PostgreSQL

[\[Notes en ligne\]](#)

Postgres, initialement développé au département de Science informatique, à UC Berkeley, mis en place la majorité des concepts des bases relationnelles, actuellement disponibles sur le marché. PostgreSQL accepte le langage SQL92/SQL3, assure l'intégrité transactionnelle, et l'extension de type. PostgreSQL est une évolution du code originale de Berkeley : il est Open Source et dans le domaine public.

PostgreSQL est disponible sans frais. La version actuelle est disponible à (en anglais) :

www.PostgreSQL.org.

Depuis la version 6.3 (03/02/1998) PostgreSQL utilise les sockets UNIX, et une table est dédiée à ces nouvelles capacités. La socket est située dans le dossier ``/tmp/.s.PGSQL.5432'`. Cette option peut être activée avec `'-i'` passé au `postmaster` et cela s'interprète: "écoute sur les sockets TCP/IP et sur les sockets Unix".

Postmaster	PHP	Statut
<code>postmaster &</code>	<code>pg_connect("", "", "", "", "dbname");</code>	OK
<code>postmaster -i &</code>	<code>pg_connect("", "", "", "", "dbname");</code>	OK
<code>postmaster &</code>	<code>pg_connect("localhost", "", "", "", "dbname");</code>	Unable to connect to PostgreSQL server: connectDB() failed: Is the postmaster running and accepting TCP/IP (with -i) connection at 'localhost' on port '5432'? in /path/to/file.php3 on line 20. : Impossible de se connecter au serveur PostgreSQL: connectDB() a échoué. Est ce que le postmaster fonctionne, et accepte les TCP/IP (option -i) sur le port '5432'? dans /path/to/file.php3, ligne 20 ? @tab
<code>postmaster -i &</code>	<code>pg_connect("localhost", "", "", "", "dbname");</code>	OK

Il est possible de se connecter avec la commande suivante : `$conn =`

```
pg_Connect("host=localhost port=5432 dbname=chris");
```

Pour utiliser l'interface des grands objets (large object (lo) interface), il est nécessaire de les placer dans un bloc de transaction. Un bloc de transaction commence avec `begin` et si la transaction se termine avec un `commit` et `end`. Si la transaction échoue, elle doit être conclue par un `abort` et `rollback`. **Utilisation des objets de grande taille (Large Objects)**

```
<?php
$database = pg_Connect ("", "", "", "", "jacarta");
pg_exec ($database, "begin");
    $oid = pg_locreate ($database);
echo (" $oid\n");
    $handle = pg_loopopen ($database, $oid, "w");
echo (" $handle\n");
pg_lowrite ($handle, "gaga");
pg_loclose ($handle);
pg_exec ($database, "commit")
pg_exec ($database, "end")
?>
```

[10.54.1 pg_Close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [pg_close](#) (int *connection*)

Retourne FALSE si l'index de connexion n'est pas valable, et TRUE sinon.
Ferme la connexion avec le serveur PostgreSQL associé à l'index de connexion.

[10.54.2 pg_cmdTuples](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_cmdtuples](#) (int *result_id*)

[pg_cmdtuples\(\)](#) retourne le nombre de tuples (instances) affectés par les requêtes INSERT, UPDATE, et DELETE. Si aucun tuple n'a été affecté, la fonction retournera 0. *pg_cmdtuples*

```
<?php
$result = pg_exec($conn, "INSERT INTO verlag VALUES ('Autor')");
$cmdtuples = pg_cmdtuples($result);
echo $cmdtuples . " <- tuples modifiés.";
?>
```

[10.54.3 pg_Connect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_connect](#) (string *host*, string *port*, string *options*, string *tty*, string *dbname*)

Retourne un index de connexion en cas de succès, et FALSE si une connexion n'a pas pu être établie. Ouvre une connexion avec un serveur PostgreSQL. Chaque argument doit être placé entre guillemets, y compris le numéro de port. Les arguments *options* et *tty* sont optionnels et peuvent être ignorés. Cette fonction retourne un index de connexion qui sera utilisé par d'autres fonctions PostgreSQL. Vous pouvez ouvrir plusieurs connexions simultanément.

Une connexion peut être aussi établie avec la commande suivante : `$conn =`

`pg_connect("dbname=marliese port=5432");` en dehors des paramètres *dbname* et *port*, *host*, *tty*, *options*, *user* et *password* sont des options.

Voir aussi [pg_pconnect\(\)](#).

[10.54.4 pg_DBname](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_dbname](#)(int *connection*)

Retourne le nom de la base de donnée PostgreSQL associée à l'index de connexion *connection*, ou FALSE si *connection* n'est pas valide.

[10.54.5 pg_ErrorMessage](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_erroremessage](#)(int *connection*)

Retourne une chaîne contenant le dernier message d'erreur, ou FALSE en cas d'échec. Il sera impossible d'obtenir des détails sur l'erreur générée, en utilisant la fonction [pg_erroremessage\(\)](#) si une erreur est survenue dans la dernière action pour laquelle une connexion valide existe. Cette fonction retournera une chaîne contenant le message d'erreur généré par le serveur final.

[10.54.6 pg_Exec](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_exec](#)(int *connection*, string *query*)

Retourne un index de résultat, si la requête a été correctement exécutée, et FALSE en cas d'échec, ou si la connexion *connection* n'était pas un index de connexion valide. En cas d'erreur, le message d'erreur peut être obtenu gr'ce à la fonction [pg_erroremessage\(\)](#), si l'index de connexion était valide. Envoie une requête à un serveur PostgreSQL identifié gr'ce à l'index de connexion. La réponse retournée par cette fonction est un index de résultat qui devra être utilisé pour accéder aux lignes de résultat, gr'ce à d'autres fonctions PostgreSQL. Note : *PHP/FI* retournait 1 lorsque la requête n'attendait pas de données en réponse (insertion, modifications, par exemple), et retournait un nombre plus grand que 1, même sur un select qui donnait un ensemble vide. Ce n'est plus le cas.

[10.54.7 pg_Fetch_Array](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [pg_fetch_array](#)(int *result*, int *row*, int *result_type*)

Retourne un tableau qui contient à la ligne demandée, dans le résultat identifiée par *result*, et FALSE, si il ne reste plus de lignes.

[pg_fetch_array\(\)](#) est une version évoluée de [pg_fetch_row\(\)](#). En plus de proposer un tableau à indice numérique, elle peut aussi enregistrer les données dans un tableau associatif, en utilisant les noms des champs comme clés.

L'argument optionnel *result_type* de [pg_fetch_array\(\)](#) est une constante, qui peut prendre les valeurs

suivantes : PGSQL_ASSOC, PGSQL_NUM, et PGSQL_BOTH. Note : **Result_type** a été ajoutée en PHP 4.0.

Il est important de noter que [pg_fetch_array\(\)](#) n'est pas significativement plus lent que [pg_fetch_row\(\)](#), tandis qu'elle fournit un confort d'utilisation notable.

Pour plus de détails, reportez vous à [pg_fetch_row\(\)](#).

PostgreSQL fetch array

```
<?php
$conn = pg_pconnect("", "", "", "", "publisher");
if (!$conn) {
    echo "Erreur de connexion.\n";
    exit;
}
$result = pg_Exec ($conn, "SELECT * FROM auteurs");
if (!$result) {
    echo "Erreur durant la requete.\n";
    exit;
}
$arr = pg_fetch_array ($result, 0);
echo $arr[0] . " <- array\n";
$arr = pg_fetch_array ($result, 1);
echo $arr["author"] . " <- array\n";
?>
```

10.54.8 pg Fetch Object

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [pg_fetch_object\(\)](#) (int *result*, int *row*, int *result_type*)

Retourne un objet dont les membres sont les champs de la ligne demandée, ou FALSE, si il n'y a plus de lignes.

[pg_fetch_object\(\)](#) est similaire à [pg_fetch_array\(\)](#), avec une différence majeure : c'est un objet qui est retourné, au lieu d'un tableau. Par conséquent, cela signifie que vous ne pouvez accéder aux membres qu'avec leur nom, et non plus leur offset (les nombres ne sont pas autorisés comme nom de membre).

L'argument optionnel *result_type* de *result_type* est une constante qui peut prendre les valeurs suivantes : PGSQL_ASSOC, PGSQL_NUM, et PGSQL_BOTH. Note : **Result_type** a été ajouté dans PHP 4.0.

Au niveau vitesse, cette fonction est aussi rapide que [pg_fetch_row\(\)](#) et presque aussi rapide que [pg_fetch_row\(\)](#) (la différence est non significative).

Voir aussi: [pg_fetch_array\(\)](#) et [pg_fetch_row\(\)](#). **Postgres fetch object**

Exemple 1. Postgres retourne un objet

```
<?php
$database = "verlag";
$db_conn = pg_connect ("localhost", "5432", "", "", $database);
if (!$db_conn): ?>
    <H1>connexion impossible à la base postgres <? echo $database ?></H1> <?
Exit;
endif;
$qu = pg_exec ($db_conn, "SELECT * FROM verlag ORDER BY autor");
```

```

$row = 0; // postgres réclame un compteur de ligne, d'autres bases ne le font pas.
while ($data = pg_fetch_object ($qu, $row)):
echo $data->autor." (" ;
echo $data->jahr ."): " ;
echo $data->titel."<BR>" ;
    $row++;
endwhile; ?>
<PRE><?php
$fields[] = Array ("autor", "Author");
$fields[] = Array ("jahr", " Year");
$fields[] = Array ("titel", " Title");
$row = 0; // postgres réclame un compteur de ligne, d'autres bases ne le font pas.
while ($data = pg_fetch_object ($qu, $row)):
echo "-----\n";
reset ($fields);
while (list (,$item) = each ($fields)):
echo $item[1].": ".$data->$item[0]."\n";
endwhile;
    $row++;
endwhile;
echo "-----\n"; ?>
</PRE> <?php
pg_freeResult ($qu);
pg_close ($db_conn);
?>

```

10.54.9 pg_Fetch_Row

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [pg_fetch_row](#) (int *result*, int *row*)

Retourne un tableau qui contient les données de la ligne demandée, ou FALSE, si il ne reste plus de lignes. [pg_fetch_row\(\)](#) lit une ligne dans le résultat associé à l'index result. La ligne est retournée sous la forme d'un tableau. La ligne est retournée sous la forme d'un tableau, qui commence à l'index 0. Les appels ultérieurs à [pg_fetch_row\(\)](#) retourneront la ligne d'après, ou bien FALSE, lorsqu'il n'y aura plus de lignes.

Voir aussi: [pg_fetch_array\(\)](#), [pg_fetch_object\(\)](#), [pg_result\(\)](#). *Postgres retourne une ligne*

```

<?php
$conn = pg_pconnect("","","","publisher");
if (!$conn) {
echo "Erreur.\n";
exit;
}
$result = pg_Exec ($conn, "SELECT * FROM authors");
if (!$result) {
echo "Erreur.\n";
exit;
}
$row = pg_fetch_row ($result, 0);
echo $row[0] . " <- row\n";
$row = pg_fetch_row ($result, 1);
echo $row[0] . " <- row\n";
$row = pg_fetch_row ($result, 2);

```

```
echo $row[1] . " <- row\n";
?>
```

[10.54.10 pg_FieldIsNull](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_fieldisnull](#)(int *result_id*, int *row*, mixed *field*)

Teste si un champs est à NULL. Retourne 0 si le champs n'est pas NULL. Retourne 1 si le champs est à NULL. Le champs peut être identifié avec son nom ou son index numérique (commencant à 0).

[10.54.11 pg_FieldName](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_fieldname](#)(int *result_id*, int *field_number*)

[pg_fieldname\(\)](#) va retourner le nom du champs qui occupe la colonne numéro *field_number* dans le résultat *result_id*. La numérotation des champs commence à 0.

[10.54.12 pg_FieldNum](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_fieldnum](#)(int *result_id*, string *field_name*)

[pg_fieldnum\(\)](#) retourne le numéro de la colonne, dont le nom est *field_name*, dans le résultat *result_id*. La numérotation des champs commence à 0. Cette fonction retournera -1 en cas d'erreur.

[10.54.13 pg_FieldPrtLen](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_fieldprtlen](#)(int *result_id*, int *row_number*, string *field_name*)

[pg_fieldprtlen\(\)](#) retourne la taille imprimée (nombre de caractères) d'une valeur donnée dans un résultat PostgreSQL. La numérotation des lignes commence à 0. Cette fonction retourne -1 en cas d'erreur.

[10.54.14 pg_FieldSize](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_fieldsize](#)(int *result_id*, int *field_number*)

[pg_fieldsize\(\)](#) retourne la taille interne de stockage d'un champs donné, en octets. Retourne -1 si la taille est variable. Retourne FALSE en cas d'erreur.

10.54.15 pg FieldType

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_fieldtype](#) (int *result_id*, int *field_number*)

[pg_fieldtype\(\)](#) retourne une chaîne contenant le type du champs donné par son index *field_number* . La numérotation des champs commence à 0.

10.54.16 pg FreeResult

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_freeresult](#) (int *result_id*)

[pg_freeresult\(\)](#) n'est vraiment utile que si vous risquez d'utiliser trop de mémoire durant votre script. La mémoire occupée par les résultats est automatiquement libérée à la fin du script. Mais, si vous êtes sûr de ne pas avoir besoin du résultat ultérieurement, vous pouvez appeler [pg_freeresult\(\)](#) avec l'index de résultat comme argument, et la mémoire sera libérée.

10.54.17 pg GetLastOid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_getlastoid](#) (int *result_id*)

[pg_getlastoid\(\)](#) sert à lire l' Oid assigné à un tuple inséré, si l'index de résultat a été obtenu avec la fonction [pg_exec\(\)](#), dont la requête était exclusivement SQL INSERT. Cette fonction retourne un entier positif si un Oid valide a été trouvé. Elle retournera -1 si une erreur est survenue, ou si la dernière commande n'était pas un INSERT.

10.54.18 pg Host

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_host](#) (int *connection_id*)

[pg_host\(\)](#) retourne le nom d'hôte associé à l'index de connexion PostgreSQL.

10.54.19 pg loclose

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pg_loclose](#) (int *fd*)

[pg_loclose\(\)](#) ferme un objet de type Inversion Large Object. *fd* est un descripteur de fichier, obtenu avec [pg_loopen\(\)](#).

10.54.20 pg locreate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_locreate](#)(int *conn*)

[pg_locreate\(\)](#) crée un objet de type Inversion Large Object et retourne son Oid. *conn* doit être une connexion valide avec une base de données PostgreSQL. Les modes d'accès PostgreSQL INV_READ, INV_WRITE, et INV_ARCHIVE ne sont pas supportés : l'objet peut toujours être créé, avec des droits d'accès en lecture et écriture. Le mode INV_ARCHIVE a été supprimé des bases PostgreSQL (version 6.3 et ultérieur).

[10.54.21 pg_loopen](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_loopen](#)(int *conn*, int *objoid*, string *mode*)

[pg_loopen\(\)](#) ouvre un objet de type Inversion Large Object et retourne un descripteur de fichier pour cet objet. Le descripteur de fichier contient les informations de connexion. Ne refermez pas la connexion avant d'avoir fermé l'objet. *objoid* est un Oid valide de Large Object, et *mode* peut prendre es valeurs suivantes : "r", "w", ou "rw".

[10.54.22 pg_loread](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_loread](#)(int *fd*, int *len*)

[pg_loread\(\)](#) lit au plus *len* octets d'un objet de grande taille, et retourne les données sous la forme d'une chaîne. *fd* est un identifiant valide d'objet de grande taille, et *len* indique la taille maximale de mémoire alloué à l'objet de grande taille.

[10.54.23 pg_loreadall](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pg_loreadall](#)(int *fd*)

[pg_loreadall\(\)](#) lit un objet de grande taille en totalité et le passe directement au client, après les entêtes adéquats. Cette fonction est prévue pour transmettre des sons ou des images.

[10.54.24 pg_lounlink](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pg_lounlink](#)(int *conn*, int *lobjid*)

[pg_lounlink\(\)](#) efface l' objet de grande taille dont l'identifiant est *lobjid*.

[10.54.25 pg_lowrite](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_lowrite](#)(int *fd*, string *buf*)

[pg_lowrite\(\)](#) écrit dans l'objet de grande taille autant de données possible, issues de la variable *buf* et retourne

le nombre d'octets réellement écrits, ou FALSE en cas d'erreur. *fd* est un descripteur d'objet de grande taille, obtenu avec [pg_loopen\(\)](#).

[10.54.26 pg_NumFields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_numfields](#)(int *result_id*)

[pg_numfields\(\)](#) retourne le nombre de champs ou (colonnes) d'un résultat PostgreSQL. L'argument doit être un identifiant de résultat valide retourné par [pg_exec\(\)](#). Cette fonction retournera -1 en cas d'erreur.

[10.54.27 pg_NumRows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_numrows](#)(int *result_id*)

[pg_numrows\(\)](#) retourne le nombre de lignes d'un résultat PostgreSQL. L'argument doit être un identifiant de résultat valide retourné par [pg_exec\(\)](#). Cette fonction retournera -1 en cas d'erreur.

[10.54.28 pg_Options](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_options](#)(int *connection_id*)

[pg_options\(\)](#) retourne une chaîne contenant les options de la connexion PostgreSQL.

[10.54.29 pg_pConnect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_pconnect](#)(string *host*, string *port*, string *options*, string *tty*, string *dbname*)

Retourne un index de connexion en cas de succès, ou FALSE en cas d'erreur. Ouvre une connexion permanente à une base PostgreSQL. Chacun des arguments doit être entre guillemets, y compris le numéro de port. Les arguments *options* et *tty* sont optionnels, et peuvent être ignorés. Cette fonction retourne un index de connexion, nécessaire aux autres fonctions PostgreSQL. Vous pouvez établir plusieurs connexion persistantes en même temps. Voir aussi [pg_connect\(\)](#).

Une connexion peut aussi être établie avec la commande suivante : `$conn = pg_pconnect("dbname=marliese port=5432");` les autres paramètres en dehors de *dbname* et *port* sont *host*, *tty*, *options*, *user* et *password* sont optionnels.

[10.54.30 pg_Port](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_port](#)(int *connection_id*)

[pg_port\(\)](#) retourne le numéro de port de la connexion identifiée *connection_id*.

10.54.31 pg_Result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [pg_result](#)(int *result_id*, int *row_number*, mixed *fieldname*)

[pg_result\(\)](#) retourne les valeurs d'un identifiant de résultat, produit par [pg_exec\(\)](#). Les arguments *row_number* et *fieldname* précisent la cellule qui sera retournée. La numérotation des lignes commence à 0. Au lieu d'utiliser le nom du champs, vous pouvez utiliser son index, sous la forme d'un nombre sans guillemets. La numérotation des champs commence à 0.

PostgreSQL dispose de nombreux types, et seuls, les types basiques sont supportés ici. Toutes les formes d'entier, booléen et Oid sont retournés sous la forme d'entiers. Toutes les formes de nombre à virgule flottante et types réels sont retournés sous la forme d'une valeur de type double. Tous les autres types, y compris les tableaux, sont retournés sous la forme de chaînes formatées, au format par défaut de PostgreSQL.

10.54.32 pg_tty

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_tty](#)(int *connection_id*)

[pg_tty\(\)](#) retourne le nom de tty de la connexion associée à *connection_id*.

10.55 Fonctions POSIX

[\[Notes en ligne\]](#)

Ce module contient une interface avec les documents au standard IEEE 1003.1 (POSIX.1), qui ne sont pas accessibles autrement. Par exemple, POSIX.1 définit les fonctions `open()`, `read()`, `write()` et `close()`, qui ont été traditionnellement les fonctions de PHP3. Certains fonctionnalités spécifiques ne sont pas encore disponibles, bien que ce module t'che de remédier à cette situation avec ses fonctions.

10.55.1 posix_kill

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [posix_kill](#)(int *pid*, int *sig*)

Envoie le signal *sig* au processus *pid*. Retourne FALSE, si il n'a pas pu envoyer le signal, et TRUE sinon. Reportez vous à la page de manuel de `kill(2)` de votre système POSIX, qui contient plus de détails sur les identifiants négatifs de processus, les pid spéciaux 0 et -1, et le signal numéro 0.

10.55.2 posix_getpid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getpid](#)(void)

Retourne l'identifiant du processus courant.

[10.55.3 posix_getppid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getppid](#)(void)

Retourne l'identifiant du processus parent du processus courant.

[10.55.4 posix_getuid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getuid](#)(void)

Retourne l'ID numérique de l'utilisateur du processus courant. Reportez vous à [posix_getpwuid\(\)](#) pour accéder au nom d'utilisateur.

[10.55.5 posix_geteuid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_geteuid](#)(void)

Retourne l'UID effectif de l'utilisateur du processus courant. Reportez vous à [posix_getpwuid\(\)](#) pour obtenir le nom d'utilisateur.

[10.55.6 posix_getgid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getgid](#)(void)

Retourne l'UID du groupe du processus courant. Reportez vous à [posix_getgrgid\(\)](#) pour accéder au nom du groupe.

[10.55.7 posix_getegid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getegid](#)(void)

Retourne l'ID effectif du groupe du processus courant. Reportez vous à [posix_getgrgid\(\)](#) pour transformer cette information en nom de groupe.

[10.55.8 posix_setuid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [posix_setuid](#)(int *uid*)

Fixe l'UID effective de l'utilisateur du processus courant. Vous devez avoir les privilèges nécessaires (traditionnellement ceux du root) sur votre système pour faire ceci.

Retourne TRUE en cas de succès, FALSE sinon. Voir aussi [posix_setgid\(\)](#).

[10.55.9 posix_setgid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [posix_setgid](#)(int *gid*)

Fixe le GID effective du processus courant. Reportez vous à [posix_getgrgid\(\)](#) pour transformer cette information en nom de groupe. L'ordre approprié est d'abord [posix_setgid\(\)](#), puis [posix_setuid\(\)](#). Retourne TRUE en cas de succès, FALSE sinon.

[10.55.10 posix_getgroups](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_getgroups](#)(void)

Retourne un tableau contenant les identifiants du groupe du processus courant. Reportez vous à [posix_getgrgid\(\)](#) pour pouvoir utiliser ces id.

[10.55.11 posix_getlogin](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [posix_getlogin](#)(void)

Retourne le nom de login de l'utilisateur qui possède le processus courant. Reportez vous à [posix_getpwnam\(\)](#) pour obtenir plus d'information sur cet utilisateur.

[10.55.12 posix_getpgrp](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getpgrp](#)(void)

Retourne l'identifiant du groupe de processus du processus courant. Reportez vous à POSIX.1 et à [getpgrp\(2\)](#) dans le manuel de votre système POSIX pour plus d'informations sur les groupes de processus.

[10.55.13 posix_setsid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_setsid](#)(void)

Fait du processus courant un chef de session. Reportez vous à POSIX.1 et [setsid\(2\)](#) dans le manuel de votre système POSIX pour plus d'informations sur le contrôle de t'che. Retourne un identifiant de session.

[10.55.14 posix_setpgid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_setpgid](#)(int *pid*, int *pgid*)

Ajoute le processus *pid* au groupe d'id *pgid*. Reportez vous à POSIX.1 et setsid(2) dans le manuel de votre système POSIX pour plus d'informations sur le contrôle de t'che. Retourne TRUE en cas de succès, et FALSE sinon.

[10.55.15 posix_getpgid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getpgid](#)(int *pid*)

Retourne l'id du groupe de processus pour le processus *pid*.

Ceci n'est pas une fonction POSIX, mais elle est répandu sur les systèmes BSD et System V. Si votre système ne supporte pas cette fonction, la fonction PHP retournera toujours FALSE.

[10.55.16 posix_getsid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getsid](#)(int *pid*)

Retourne le sid du processus *pid*. Si *pid* est à 0, le sid retourné sera celui du processus courant.

Ceci n'est pas une fonction POSIX, mais elle est répandu sur les systèmes BSD et System V. Si votre système ne supporte pas cette fonction, la fonction PHP retournera toujours FALSE.

[10.55.17 posix_uname](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_uname](#)(void)

Retourne un tableau associatif avec des informations sur le système. Les indices du tableau sont :

- sysname – nom du système d'exploitation (e.g. Linux)
- nodename – nom du système (e.g. valiant)
- release – édition du système d'exploitation (e.g. 2.2.10)
- version – version du système d'exploitation (e.g. #4 Tue Jul 20 17:01:36 MEST 1999)
- machine – architecture système (e.g. i586)

Posix impose que vous n'avez pas d'a priori sur le format des chaînes, c'est à dire que vous ne devez pas vous attendre à avoir forcément 3 chiffres pour la version, par exemple.

[10.55.18 posix_times](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_times](#)(void)

Retourne un tableau avec les informations sur l'utilisation du CPU. Les indices sont :

- ticks – nombre de ticks depuis le dernier démarrage
- utime – temps utilisateur utilisé par le processus courant.
- stime – temps système utilisé par le processus courant.
- cutime – temps utilisateur utilisé par le processus courant et ses enfants.
- cstime – temps système utilisé par le processus courant et ses enfants.

[10.55.19 posix_ctermid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [posix_ctermid](#)(void)

Encore à faire.

[10.55.20 posix_ttyname](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [posix_ttyname](#)(int *fd*)

Encore à faire.

[10.55.21 posix_isatty](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [posix_isatty](#)(int *fd*)

Encore à faire.

[10.55.22 posix_getcwd](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [posix_getcwd](#)(void)

Encore à faire très rapidement.

[10.55.23 posix_mkfifo](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [posix_getcwd](#)(string *pathname*, int *mode*)

Encore à faire très rapidement.

[10.55.24 posix_getgrnam](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_getgrnam](#) (string *name*)

Encore à faire.

[10.55.25 posix_getgrgid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_getgrgid](#) (int *gid*)

Encore à faire.

[10.55.26 posix_getpwnam](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_getpwnam](#) (string *username*)

Retourne un tableau associatif qui contient des informations à propos d'un utilisateur, identifié par son nom, passé en paramètre *username*.

Les éléments du tableau sont :

Élément	Description
name	Le nom contient le nom de l'utilisateur. Généralement, c'est un nom court, de moins de 16 caractères, mais ce n'est pas son nom réel et complet. Cette valeur devrait correspondre au paramètre <i>username</i> , et donc, il est redondant. @tab
passwd	Contient le mot de passe de l'utilisateur, encrypté. Souvent, dans les systèmes utilisant les mots de passes "fantômes", un astérisque est retourné. @tab
uid	L'UID de l'utilisateur. @tab
gid	L'ID du groupe de l'utilisateur. Utilisez la fonction posix_getgrgid() pour connaître le nom du groupe, et ses membres. @tab
gecos	GECOS est un terme obsolète qui fait référence aux données de finger, sur un système Honeywell. Le champs, cependant, a survécu, et son contenu a été formalisé par POSIX. Le champs contient une liste, séparée par des virgules, qui contient le nom complet de l'utilisateur, son téléphone professionne, son numéro de bureau, et son numéro de téléphone personnel. Sur la plus part des sytèmes, seul le nom est disponible. @tab

dir	Cet élément contient le chemin absolu jusqu'au dossier racine de l'utilisateur. @tab
shell	Cet élément contient le chemin absolu jusqu'au dossier d'exécution du shell de l'utilisateur. @tab

[10.55.27 posix_getpwuid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_getpwuid](#) (int *uid*)

Retourne un tableau associatif contenant des informations sur un utilisateur repéré par son UID, passé dans le paramètre *uid*.

Les éléments du tableau sont :

Elément	Description
name	Le nom contient le nom de l'utilisateur. Généralement, c'est un nom court, de moins de 16 caractères, mais ce n'est pas son nom réel et complet. @tab
passwd	Contient le mot de passe de l'utilisateur, encrypté. Souvent, dans les systèmes utilisant les mots de passes "fantômes", un astérisque est retourné. @tab
uid	Cette valeur devrait correspondre au paramètre <i>uid</i> , et donc, il est redondant. @tab
gid	L'ID du groupe de l'utilisateur. Utilisez la fonction posix_getgrgid() pour connaître le nom du groupe, et ses membres. @tab
gecos	GECOS est un terme obsolète qui fait référence aux données de finger, sur un système Honeywell. Le champs, cependant, a survécu, et son contenu a été formalisé par POSIX. Le champs contient une liste, séparée par des virgules, qui contient le nom complet de l'utilisateur, son téléphone professionnelle, son numéro de bureau, et son numéro de téléphone personnel. Sur la plus part des systèmes, seul le nom est disponible. @tab
dir	Cet élément contient le chemin absolu jusqu'au dossier racine de l'utilisateur.

	@tab
shell	Cet élément contient le chemin absolu jusqu'au dossier d'exécution du shell de l'utilisateur. @tab

[10.55.28 posix_getrlimit](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_getrlimit](#)(void)

Encore à faire ASAP.

[10.56 Pspell Functions](#)

[\[Notes en ligne\]](#)

The `@xref{function.pspell , , pspell()}` functions allows you to check the spelling on a word and offer suggestions.

You need the aspell and pspell libraries, available from: <http://pspell.sourceforge.net/>.

[10.56.1 pspell_new](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_new](#)(string *language*, string *spelling* , string *jargon* , string *encoding*)

[pspell_new\(\)](#) opens up a new dictionary and returns the dictionary link identifier for use in other pspell functions.

The language parameter is the language code which consists of the two letter ISO 639 language code and an optional two letter ISO 3166 country code after a dash or underscore. The spelling parameter is the requested spelling for languages with more than one spelling such as English. Known values are ``american'', ``britsh'', and ``canadian''. The jargon parameter contains extra information two distinguish two different words lists that have the same language-tag and spelling. The encoding parameter is the encoding that words are expected to be in. Valid values are 'utf-8', 'iso8859-*', 'koi8-r', 'viscii', 'cp1252', 'machine unsigned 16', 'machine unsigned 32'. This parameter is largely untested, so be careful when using. For more information and examples, check out inline manual pspell website: <http://pspell.sourceforge.net/>.

[pspell_new\(\)](#)

```
$pspell_link = pspell_new ("english");
```

10.56.2 pspell_mode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [pspell_mode](#)(int *dictionary_link*, int *mode*)

[pspell_mode\(\)](#) changes the spellchecking mode.

There are three modes available:

- PSPELL_FAST – Fast mode (least number of suggestions)
- PSPELL_NORMAL – Normal mode (more suggestions)
- PSPELL_BAD_SPELLERS – Slow mode (a lot of suggestions)

[pspell_mode\(\)](#)

```
$pspell_link = pspell_new ("english");
pspell_mode (PSPELL_FAST);
if (!pspell_check ($pspell_link, "testt")) {
    $suggestions = pspell_suggest ($pspell_link, "testt");
}
```

10.56.3 pspell_runtogether

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [pspell_runtogether](#)(int *dictionary_link*, int *mode*)

[pspell_runtogether\(\)](#) Consider run-together words as legal compounds. That is, "thecat" will be a legal compound, although there should be a space between the two words. Changing this setting only affects the results returned by [pspell_check\(\)](#); [pspell_suggest\(\)](#) will still return suggestions, and they are not affected by the calls to `@xref{function.php-runtogether, , php_runtogether()}`.

[pspell_runtogether\(\)](#)

```
$pspell_link = pspell_new ("english");
pspell_runtogether (true);
echo pspell_runtogether ($pspell_link, "thecat") ? "correct" : "wrong";
```

10.56.4 pspell_check

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [pspell_check](#)(int *dictionary_link*, string *word*)

[pspell_check\(\)](#) checks the spelling of a word and returns true if the spelling is correct, false if not.

[pspell_check\(\)](#)

```
$pspell_link = pspell_new ("english");
if (pspell_check ($pspell_link, "testt")) {
echo "This is a valid spelling";
} else {
echo "Sorry, wrong spelling";
}
```

[10.56.5 pspell_suggest](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)array [pspell_suggest](#)(int *dictionary_link*, string *word*)[pspell_suggest\(\)](#) returns an array of possible spellings for the given word.[pspell_suggest\(\)](#)

```
$pspell_link = pspell_new ("english");
if (!pspell_check ($pspell_link, "testt")){
    $suggestions = pspell_suggest ($pspell_link, "testt");
    for ($i=0; $i < count ($suggestions); $i++) {
    echo "Possible spelling: " . $suggestions[$i] . "<br>";
    }
}
```

[10.57 GNU Readline](#)

[\[Notes en ligne\]](#)

The [readline\(\)](#) functions implement an interface to the GNU Readline library. These are functions that provide editable command lines. An example being the way Bash allows you to use the arrow keys to insert characters or scroll through command history. Because of the interactive nature of this library, it will be of little use for writing Web applications, but may be useful when writing scripts meant to be run from a shell. The home page of the GNU Readline project is <http://cnswww.cns.cwru.edu/~chet/readline/rltop.html>. It's maintained by Chet Ramey, who's also the author of Bash.

[10.57.1 readline](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)string [readline](#)(string *prompt*)

This function returns a single string from the user. You may specify a string with which to prompt the user. The line returned has the ending newline removed. You must add this line to the history yourself using [readline_add_history\(\)](#).

[readline\(\)](#)

```
//get 3 commands from user
for ($i=0; $i < 3; $i++) {
    $line = readline ("Command: ");
    readline_add_history ($line);
}
//dump history
print_r (readline_list_history());
//dump variables
print_r (readline_info());
```

[10.57.2 readline_add_history](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [readline_add_history](#)(string *line*)

This function adds a line to the command line history.

[10.57.3 readline_clear_history](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [readline_clear_history](#)(void)

This function clears the entire command line history.

[10.57.4 readline_completion_function](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [readline_completion_function](#)(string *line*)

This function registers a completion function. You must supply the name of an existing function which accepts a partial command line and returns an array of possible matches. This is the same kind of functionality you'd get if you hit your tab key while using Bash.

[10.57.5 readline_info](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [readline_info](#)(string *varname* , string *newvalue*)

If called with no parameters, this function returns an array of values for all the setting readline uses. The elements will be indexed by the following values: done, end, erase_empty_line, library_version, line_buffer, mark, pending_input, point, prompt, readline_name, and terminal_name.

If called with one parameter, the value of that setting is returned. If called with two parameters, the setting will be changed to the given value.

[10.57.6 readline_list_history](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [readline_list_history](#) (void)

This function returns an array of the entire command line history. The elements are indexed by integers starting at zero.

[10.57.7 readline_read_history](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [readline_read_history](#) (string *filename*)

This function reads a command history from a file.

[10.57.8 readline_write_history](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [readline_write_history](#) (string *filename*)

This function writes the command history to a file.

[10.58 Fonction GNU Recode](#)

[\[Notes en ligne\]](#)

Ce module contient l'interface à la librairie GNU Recode library, version 3.5. Pour pouvoir utiliser ces fonctions, il faut que PHP ait été compilé avec l'option `--with-recode`. Pour cela, il faut que vous ayez la librairie GNU Recode 3.5 ou plus récent, installée sur votre système.

La librairie GNU Recode library convertit les fichiers ayant des jeux de caractères différents. Lorsque ce n'est pas possible, elle se débarrasse des caractères illégaux, ou bien effectue une approximation. La librairie reconnaît ou produit près de 150 jeux de caractères différents, et peut quasiment tous les convertir de l'un vers l'autre. La plus part des jeux de caractères de la RFC 1345 sont supportés.

[10.58.1 recode_string](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [recode_string](#) (string *request*, string *string*)

Recode la chaîne *string* en fonction de la requête *request*. Retourne FALSE, en cas d'échec, et TRUE sinon. Une requête simple de recodage peut être "lat1..iso646-de". Reportez vous à la documentation GNU Recode de votre installation pour plus de détails sur les requêtes.

[10.58.2 recode_file](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [recode_file](#) (int *input*, int *output*)

Recode le fichier identifié par *input* dans le fichier identifié par *output* en fonction de la requête de recodage *request*. Retourne FALSE, en cas d'échec, et TRUE sinon.

Cette fonction ne gère pas encore les fichiers distants (URLs). Les deux fichiers doivent faire référence à des fichiers locaux.

10.59 Expressions régulières

[\[Notes en ligne\]](#)

Les expressions régulières sont utilisées pour effectuer des manipulations complexes de chaînes de caractères. Les fonctions sont :

- [ereg\(\)](#)
- [ereg_replace\(\)](#)
- [eregi\(\)](#)
- [eregi_replace\(\)](#)
- [split\(\)](#)

Ces fonctions requièrent toutes une expression régulière comme premier argument. PHP utilise les expressions régulières avancées de POSIX (POSIX 1003.2). Pour avoir tous les détails sur ces expressions, reportez vous aux pages de manuel incluses dans le répertoire de la distribution PHP.

Expressions régulières

```
ereg("abc",$string);
/* Retourne TRUE si "abc"
est trouvé quelque part dans la chaîne $string. */
ereg("^abc",$string);
/* Retourne TRUE si "abc"
est trouvé au début de la chaîne $string. */
ereg("abc$", $string);
/* Retourne TRUE si "abc"
est trouvé à la fin de la chaîne $string. */
eregi("ozilla.[23]|MSIE.3",$HTTP_USER_AGENT);
/* Retourne TRUE si le client
est Netscape 2, 3 ou MSIE 3. */
ereg("([:alnum:]+) ([[:alnum:]]+) ([[:alnum:]]+)",
    $string,$regs);
/* Introduit trois mots séparés par des espaces
dans les chaînes $regs[1], $regs[2] et $regs[3]. */
$string = ereg_replace("^","<BR>",$string);
/* Insère une balise <BR> au début de la chaîne $string. */
$string = ereg_replace("$","<BR>",$string);
/* Insère une balise <BR> à la fin de la chaîne $string. */
$string = ereg_replace("\n","", $string);
/* Supprime toutes les nouvelles lignes de $string. */
```

10.59.1 ereg

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ereg](#)(string *pattern*, string *string*, array *regs*)

Recherche dans la chaîne *string* les séquences de caractères qui correspondent au masque *pattern*. Si au moins une séquence est trouvée (éventuellement dans les parenthèses capturantes de *pattern*), et que la fonction est appelée avec un troisième argument *regs*, les résultats seront enregistrés dans *regs*. `$regs[1]` contiendra la première parenthèse capturante (celle qui commence le plus tôt), `$regs[2]` contiendra la deuxième parenthèse capturante (celle qui commence après la première), et ainsi de suite. `$regs[0]` contient une copie de la chaîne.

La recherche est sensible à la casse.

Retourne TRUE si une occurrence a été trouvée dans la chaîne, et FALSE dans le cas contraire, ou si une erreur est survenue.

L'exemple suivant prend une date au format ISO (YYYY-MM-DD) et l'affiche sous la forme DD.MM.YYYY : *Exemple ereg()*

```
if ( ereg( "[0-9]{4})-([0-9]{1,2})-([0-9]{1,2})", $date, $regs ) ) {
echo "$regs[3].$regs[2].$regs[1]";
} else {
echo "Format de date invalide : $date";
}
```

Voir aussi [ereg\(\)](#), [ereg_replace\(\)](#) et [eregi_replace\(\)](#).

10.59.2 [ereg_replace](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ereg_replace](#)(string *pattern*, string *replacement*, string *string*)

Cette fonction effectue une recherche par expression régulière dans la chaîne *string* en recherchant les occurrences de *pattern*, puis les remplace par la chaîne *replacement*.

La chaîne modifiée est retournée. (Ce qui signifie que la chaîne originale sera retournée si aucune occurrence n'est trouvée).

Si *pattern* contient des parenthèses capturantes, *replacement* pourra contenir des séquences de la forme `\\digit`, qui seront remplacées par le texte capturé par la n-ième parenthèse capturante. `\\0` correspond à la chaîne originale complète. De 0 à 9 parenthèses capturantes peuvent être utilisées. Les parenthèses peuvent être imbriquées, et leur numéro d'ordre est défini par leur parenthèse ouvrante.

Si aucune occurrence n'est trouvée, la chaîne *string* sera retournée intacte.

Par exemple, le code suivant affiche "Ceci est un test" trois fois : *Exemple avec [ereg_replace\(\)](#)*

```
$string = "Ceci est un test";
echo ereg_replace( " est", " etait", $string );
echo ereg_replace( "( )est ", "\\1etait", $string );
echo ereg_replace( "(( )est)", "\\2etait", $string );
```

Voir aussi [ereg\(\)](#), [eregi\(\)](#) et [eregi_replace\(\)](#).

10.59.3 eregi

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [eregi](#)(string *pattern*, string *string*, array *regs*)

Cette fonction est identique à [ereg\(\)](#), hormis le fait qu'elle ignore la casse des caractères lors de la recherche sur les caractères alphabétiques.

Voir aussi [ereg\(\)](#), [ereg_replace\(\)](#) et [eregi_replace\(\)](#).

10.59.4 eregi_replace

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [eregi_replace](#)(string *pattern*, string *replacement*, string *string*)

Cette fonction est identique à [ereg_replace\(\)](#), hormis le fait qu'elle ne tient pas compte de la casse des caractères alphabétiques.

Voir aussi [ereg\(\)](#), [eregi\(\)](#) et [ereg_replace\(\)](#).

10.59.5 split

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [split](#)(string *pattern*, string *string*, int *limit*)

Retourne un tableau de chaînes : chacune d'entre elles est une sous-chaîne de *string* délimitée par les occurrences trouvées de l'expression régulière *pattern*. Si une erreur survient, retourne FALSE.

Pour lire les 5 premiers champs d'une ligne du fichier `/etc/passwd` : *Exemple avec [split\(\)](#)*

```
$passwd_list = split( ":", $passwd_line, 5 );
```

Pour analyser une date qui est délimitée par des /, des points ou des tirets : *Exemple avec [split\(\)](#)*

```
$date = "04/30/1973"; // Les délimiteurs peuvent être des /, des points ou des tirets
list( $month, $day, $year ) = split( '[/.-]', $date );
echo "Mois: $month; Jour: $day; Année: $year<br>\n";
```

Notez que *pattern* est insensible à la casse

Notez bien que si vous n'avez pas besoin de la puissance des expressions régulières, il est plus rapide d'utiliser [explode\(\)](#), qui n'utilise pas le moteur d'expressions régulières.

Notez aussi que *pattern* est une expression régulière. Si vous voulez utiliser n'importe quel caractère spécial des expressions régulières, vous devez les échapper. Si vous pensez que [split\(\)](#) (ou toute autre expression régulière) se comporte bizarrement, lisez d'abord le fichier `regex.7`, inclus dans le dossier `regex/` de la distribution PHP. Il est au format manpage, et vous pourrez le lire avec une commande telle que `man /usr/local/src/regex/regex.7`.

Voir aussi : [explode\(\)](#) et [implode\(\)](#).

10.59.6 sql_regcase

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [sql_regcase](#) (string *string*)

Retourne une expression régulière valide qui acceptera la chaîne *string*, et toutes les variantes majuscule/minuscule possibles de cette chaîne. Cette expression sera construite à partir de la chaîne *string* en remplaçant tous les caractères par des expressions entre crochets (des classes de caractères), contenant la lettre majuscule et minuscule. Si le caractère n'est pas une lettre, les crochets contiendront deux fois le caractère original. *Exemple avec [sql_regcase\(\)](#)*

```
echo sql_regcase( "Foo bar" );
```

affichera @exemple [Ff][Oo][Oo][] [Bb][Aa][Rr] .

Cette expression sert à effectuer des recherches insensibles à la casse avec d'autres logiciels, qui n'acceptent les recherches insensibles à la casse.

10.60 Satellite CORBA client extension

[\[Notes en ligne\]](#)

The Satellite extension is used for accessing CORBA objects. You will need to set the `idl_directory=` entry in `php.ini` to a path where you store all IDL files you use.

10.60.1 OrbitObject

[\[Notes en ligne\]](#)

new @xref{function.orbitobject , , orbitobject (string *ior*) }

This class provides access to a CORBA object. The *ior* parameter should be a string containing the IOR (Interoperable Object Reference) that identifies the remote object.

Sample IDL file

```
interface MyInterface {
void SetInfo (string info);
string GetInfo();
attribute int value;
}
```

PHP code for accessing MyInterface

```
<?php
$obj = new OrbitObject ($ior);
$obj->SetInfo ("A 2Good object");
```

```
echo $obj->GetInfo();
$obj->value = 42;
echo $obj->value;
?>
```

10.60.2 OrbitEnum

[\[Notes en ligne\]](#)

```
new @xref{function.orbitenum , , orbitenum (string id) }
```

This class represents the enumeration identified with the *id* parameter. The *id* can be either the name of the enumeration (e.g. "MyEnum"), or the full repository id (e.g. "IDL:MyEnum:1.0").

Sample IDL file

```
enum MyEnum {
a,b,c,d,e
};
```

PHP code for accessing MyEnum

```
<?php
$enum = new OrbitEnum ("MyEnum");
echo $enum->a; /* write 0 */
echo $enum->c; /* write 2 */
echo $enum->e; /* write 4 */
?>
```

10.60.3 OrbitStruct

[\[Notes en ligne\]](#)

```
new @xref{function.orbitstruct , , orbitstruct (string id) }
```

This class represents the structure identified with the *id* parameter. The *id* can be either the name of the struct (e.g. "MyStruct"), or the full repository id (e.g. "IDL:MyStruct:1.0").

Sample IDL file

```
struct MyStruct {
short shortvalue;
string stringvalue;
};
interface SomeInterface {
void SetValues (MyStruct values);
```

```
MyStruct GetValues();
}
```

PHP code for accessing MyStruct

```
<?php
$obj = new OrbitObject ($ior);
$initial_values = new OrbitStruct ("IDL:MyStruct:1.0");
$initial_values->shortvalue = 42;
$initial_values->stringvalue = "HGTG";
$obj->SetValues ($initial_values);
$values = $obj->GetValues();
echo $values->shortvalue;
echo $values->stringvalue;
?>
```

10.60.4 satellite_caught_exception

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [satellite_caught_exception](#),

This function returns true if an exception has been caught.

Sample IDL file

```
/* +++?????++ Out of Cheese Error. Redo From Start. */
exception OutOfCheeseError {
int parameter;
}
interface AnotherInterface {
void AskWhy() raises (OutOfCheeseError);
}
```

PHP code for handling CORBA exceptions

```
<?php
$obj = new OrbitObject ($ior);
$obj->AskWhy();
if (satellite_caught_exception()) {
if ("IDL:OutOfCheeseError:1.0" == satellite_exception_id()) {
    $exception = satellite_exception_value();
echo $exception->parameter;
}
}
?>
```

[10.60.5 satellite_exception_id](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [satellite_exception_id](#),

Return a repository id string. (E.g. "IDL:MyException:1.0".) For example usage see [satellite_caught_exception\(\)](#).

[10.60.6 satellite_exception_value](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

OrbitStruct [satellite_exception_value](#),

Return an exception struct. For example usage see [satellite_caught_exception\(\)](#).

[10.61 Sémaphores et gestion de la mémoire partagée](#)

[\[Notes en ligne\]](#)

Ce module fourni un système de sémaphore. Ce système utilise les sémaphores System V. les sémaphores peuvent être utilisés pour fournir un accès exclusif à certaines ressources de la machine, ou pour limiter le nombre de processus qui utilisent en même temps une ressource.

Ce module fournit aussi un système de mémoire partagée, qui utilise la mémoire partagée System V. Cette mémoire partagée permet d'accéder à des variables globales. Les différents démons httpd et mêmes d'autres programmes (tels que Perl, C, ...) permettent un tel échange de données global. N'oubliez pas que la mémoire partagée n'est pas protégées contre l'accès simultanée. Il vous faudra utiliser les sémaphores pour assurer la synchronisation.

SHMMAX	Taille maximale de mémoire partagée, par défaut, 131072 octets.
SHMMIN	Taille minimale de mémoire partagée, par défaut, 1 octet.
SHMMNI	Nombre maximal de segment de mémoire partagée, par défaut 100.
SHMSEG	Taille maximale de mémoire partagée par processus, par défaut 6.

[10.61.1 sem_get](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sem_get](#)(int *key*, int *max_acquire* , int *perm*)

Retourne un identifiant positif de sémaphore en cas de succès, et FALSE en cas d'erreur.

[sem_get\(\)](#) retourne un identifiant qui pourra être utilisé pour accéder à un sémaphore System V. Le sémaphore est créé, si nécessaire, en utilisant les bits de permission (par défaut, 0666). Le nombre de processus qui peuvent réserver simultanément le sémaphore est précisé dans *max_acquire* (par défaut à 1). Actuellement, cette valeur n'est affectée que si le processus est le seul processus actuellement attaché au sémaphore.

Un deuxième appel à [sem_get\(\)](#) avec la même clé retournera un identifiant différent, mais les deux identifiants permettront d'accéder au même sémaphore.

Voir aussi : [sem_acquire\(\)](#) et [sem_release\(\)](#).

10.61.2 sem_acquire

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sem_acquire](#)(int *sem_identifiant*)

Retourne TRUE en cas de succès, et FALSE sinon.

[sem_acquire\(\)](#) se bloque (si nécessaire) jusqu'à ce que le sémaphore puisse être réservé. Un processus qui tente de réserver un sémaphore qu'il a déjà réservé restera en attente indéfinie, si cette acquisition excède le nombre `max_acquire` de réservation simultanée.

A la fin d'un script, tous les sémaphores réservés mais non explicitement libérés seront libérés automatiquement, et une alerte sera générée.

Voir aussi : [sem_get\(\)](#) et [sem_release\(\)](#).

10.61.3 sem_release

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sem_release](#)(int *sem_identifiant*)

Retourne TRUE en cas de succès, FALSE en cas d'erreur.

[sem_release\(\)](#) libère le sémaphore s'il a été réservé par le processus courant. Sinon, génère une erreur.

Après libération du sémaphore, [sem_acquire\(\)](#) peut être appelé pour le réserver à nouveau.

Voir aussi : [sem_get\(\)](#) et [sem_acquire\(\)](#).

10.61.4 shm_attach

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_attach](#)(int *key*, int *memsize*, int *perm*)

[shm_attach\(\)](#) retourne un identifiant qui permettra d'accéder au System V de mémoire partagée. Au premier appel, la mémoire sera créée, avec la taille `mem_size` (par défaut: `sysvshm.init_mem` dans `php3.ini`, sinon 10000 octets) et avec les permissions `perm`(par défaut : 666).

Aux appels suivants avec la même clé *key*, [shm_attach\(\)](#) retournera un nouvel identifiant, mais cet identifiant accédera toujours à la même portion de mémoire partagée. Dans ce cas, *memsize* et *perm* seront ignorés.

10.61.5 shm_detach

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_detach](#)(int)

[shm_detach\(\)](#) rel'che le segment de mémoire partagée identifié par *shm_identifiant* et créé par [sem_get\(\)](#). N'oubliez pas que cette mémoire partagée existe toujours sous Unix, et que les données sont toujours accessibles.

10.61.6 shm_remove

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_remove](#)(int *shm_identifiant*)

Supprime un segment de mémoire partagée sous Unix. Toutes les données seront supprimées.

10.61.7 shm_put_var

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_put_var](#) (int *shm_identifieur*, int *variable_key*, mixed *variable*)

Insère ou modifie la variable *variable* avec la clé *variable_key*. Tous les types de variables (double, int, string, array) sont supportés.

10.61.8 shm_get_var

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [shm_get_var](#) (int *id*, int *variable_key*)

[shm_get_var\(\)](#) retourne la variable repérée par *variable_key*. La variable est toujours présente en mémoire partagée.

10.61.9 shm_remove_var

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_remove_var](#) (int *id*, int *variable_key*)

[shm_remove_var\(\)](#) efface la variable *variable_key* de la mémoire partagée et libère la mémoire.

10.62 Gestion des sessions

[\[Notes en ligne\]](#)

La gestion des sessions avec PHP est un moyen de sauver des informations entre deux accès. Cela permet notamment de construire des applications personnalisées, et d'accroître l'attrait de votre site.

Si vous connaissez déjà la gestion des sessions avec phplib, vous remarquerez que certains concepts sont similaires.

Chaque visiteur qui accède à votre site se voit assigner un numéro d'identifiant, appelé plus loin "identifiant de session". Celui ci est enregistré soit dans un cookie, chez le client, soit dans l'URL.

Les sessions vous permettront d'enregistrer des variables, pour les préserver et les réutiliser tout au long des requêtes. Lorsqu'un visiteur accède à votre site, PHP vérifiera automatiquement (si `session.auto_start` est à 1) ou manuellement (explicitement avec [session_start\(\)](#) ou implicitement avec [session_register\(\)](#)) si une session a déjà été ouverte. Si une telle session existe déjà, l'environnement précédent sera recréé.

Toutes les variables à enregistrer seront enregistrées sur le disque à la fin de chaque requête. Les variables enregistrées mais non définies seront marquées comme tel. Lors des accès ultérieurs, elles ne seront définies que si l'utilisateur le fait.

Les options `track_vars` et `gpc_globals` modifient la façon avec laquelle les variables sont rechargées. Si `track_vars` est activée, alors les variables de session seront accessibles dans le tableau associatif global `$HTTP_STATE_VARS`. Si `gpc_globals` est activé, alors les variables de session seront placés dans les variables globales associées. Si les deux options sont activées, alors les variables globales et `$HTTP_STATE_VARS` contiendront les valeurs de session.

Il y a deux modes de propagation de l'identifiant de session :

- Cookies
- Paramètre URL

Le module de session supporte les deux techniques. La méthode par cookies est optimale, mais étant donné le peu de fiabilité (les clients peuvent les refuser, ou les effacer), on ne peut pas se contenter de cette technique. La deuxième méthode place l'identifiant de session directement dans l'URL.

PHP est capable de gérer ceci de manière transparente, lorsque vous le compilez avec l'option `--enable-trans-sid`. Dans ce cas, les URL relatives seront modifiées pour contenir l'identifiant de session automatiquement. Sinon, vous pouvez toujours utiliser la constante `SID`, qui sera définie si le client n'envoie pas le cookie approprié. `SID` prend la forme de `session_name=session_id`, ou bien, c'est une chaîne vide. L'exemple suivant montre comment enregistrer une variable, et comment relier correctement des pages avec `SID`. *Compter le nombre de hit d'un utilisateur.*

```
<?php
session_register("compteur");
$count++;
?>
Salut visiteur, vous avez vu cette page <? echo $compteur; ?> times.<p>
<php?
# le <?=SID?> est nécessaire pour transmettre l'identifiant de session
# au cas où les utilisateurs auraient inactivé les cookies
?>
Pour continuer, <A HREF="nextpage.php?<?=SID?>">clique ici</A>
```

Pour enregistrer ces informations dans une base de données, il vous faut utiliser la fonction `session_set_save_handler` (NDtraducteur : cette fonction semble avoir disparue). Il faudra alors implémenter la fonction suivante pour l'adapter à MySQL ou toute autre base de données :

utilisation de `@xref{function.session-set-save-handler, , session_set_save_handler()}` *utilisation de*
`@xref{function.session-set-save-handler, , session_set_save_handler() }`

```
<?php
function open ($save_path, $session_name) {
echo "ouvre ($save_path, $session_name)\n";
return true;
}
function close() {
echo "ferme\n";
return true;
}
function read ($key) {
echo "écriture ($key, $val)\n";
return "foo|i:1;";
}
function write ($key, $val) {
echo "écriture ($key, $val)\n";
return true;
}
function destroy ($key) {
return true;
}
function gc ($maxlifetime) {
```

```

return true;
}
session_set_save_handler ("open", "close", "read", "write", "destroy", "gc");
session_start();
$foo++;
?>

```

Cela va produire le résultat suivant :

```

$ ./php save_handler.php
Content-Type: text/html
Set-cookie: PHPSESSID=f08b925af0ecb52bdd2de97d95cdbe6b
open (/tmp, PHPSESSID)
read (f08b925af0ecb52bdd2de97d95cdbe6b)
write (f08b925af0ecb52bdd2de97d95cdbe6b, foo|i:2;)
close

```

Le `<?=SID?>` n'est pas nécessaire, si l'option `--enable-trans-sid` a été utilisé pour compiler PHP.

Le système de gestion des sessions dispose d'un grand nombre d'options, qui sont placées dans le fichier `php.ini` file. En voici un survol rapide :

- `session.save_handler` `session.save_handler` défini les noms des fonctions qui seront utilisées pour enregistrer et retrouver les données associées à une sessions. Par défaut, les sessions sont enregistrées dans des fichiers.
- `session.save_path` défini l'argument qui est passé à la fonction de sauvegarde. Si vous utilisez la sauvegarde par fichier, cet argument est le chemin jusqu'au dossier où les fichiers sont créés. Par défaut, le dossier est `/tmp`.
- `session.name` spécifie le nom de la session, qui sera utilisé comme nom de cookie. Par défaut : `PHPSESSID`.
- `session.auto_start` indique qu'une session doit commencer automatiquement lors de la premier requête. Par défaut à 0 (inactivé).
- `session.lifetime` fixe la durée de vie, en secondes, du cookie envoyé au client. La valeur 0 signifie "jusqu'à ce que le client soit fermé". Par défaut à 0 (inactivé).
- `session.serialize_handler` défini le nom de la fonction qui sera utilisée pour enregistrer et relire les donnés. Actuellement, c'est un format interne de PHP (nom : `php`) et `WDDX` (nom : `wddx`). `WDDX` n'est utilisable que si PHP a été compilé avec le [10.73 WDDX functions](#). Par défaut, c'est le mode `php` qui est sélectionné.
- `session.gc_probability` précise la probabilité que la routine `gc` (garbage collection) soit lancée, en pourcentage. Par défaut, 1.
- `session.gc_maxlifetime` fixe la durée, en secondes, au-delà de laquelle les données considérées comme inutiles seront supprimées.
- `session.referer_check` détermine si l'identifiant de session ids utilisé par des sites externe seront éliminé. Si les identifiants de sessions sont propagés avec la méthode des URL, des utilisateurs qui n'en connaîtrait pas l'utilité risque de divulguer ces valeurs, et cela mènera à des problèmes de sécurité. Cette option y remédie. Par défaut : 0.
- `session.entropy_file` est le chemin jusqu'à une source externe (fichier) d'entropie, qui sera utilisée lors de la création de l'identifiant de session. Par exemple, `/dev/random` ou `/dev/urandom` qui sont disponibles sur de nombreux systèmes UNIX.
- `session.entropy_length` précise le nombre d'octets qui seront lus dans le fichier ci-dessus. Par défaut, 0 (inactivé).
- `session.use_cookies` `cookies` indique si le module doit utiliser des cookies pour enregistrer

l'identifiant de session chez le client. Par défaut, 1 (activé).

Note : *La gestion des sessions a été ajoutée dans PHP 4.0.*

[10.62.1 session_start](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [session_start](#)

[session_start\(\)](#) crée une session (ou continue la session courante, en fonction de l'identifiant de session passé par une variable GET ou par un cookie)

Cette fonction retourne toujours true.

Note : *Cette fonction a été ajoutée dans PHP 4.0.*

[10.62.2 session_destroy](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [session_destroy](#)

[session_destroy\(\)](#) détruit toutes les données associées à la session courante.

Cette fonction retourne toujours true.

Note : *Cette fonction a été ajoutée dans PHP 4.0.*

[10.62.3 session_name](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [session_name](#) (string *name*)

[session_name\(\)](#) retourne le nom de la session courante. Si *name* est fourni, le nom de la session changera, et prendra la valeur fournie.

Le nom de session fait référence à l'identifiant de session dans les cookies. Il ne doit contenir que des caractères alphanumériques; il doit être court et descriptif. (i.e. surtout pour les utilisateurs d'alertes de cookie). Le nom de session est remis à une valeur par défaut, enregistrées dans session.name au moment du démarrage. Ainsi, vous devez appeler [session_name\(\)](#) à chaque requête (et avant [session_start\(\)](#) ou [session_register\(\)](#)).

Exemple avec [session_name\(\)](#)

```
<?php
# Change le nom de la session à WebsiteID
$previous_name = session_name ("WebsiteID");
echo "L'ancien nom de la session était $previous_name<p>";
?>
```

Note : *Cette fonction a été ajoutée dans PHP 4.0.*

10.62.4 session_module_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [session_module_name](#)(string *module*)

[session_module_name\(\)](#) affecte et/ou retourne le module courant de session courante. Si *module* est fourni, ce module sera utilisé à la place du courant. Note : *Cette fonction a été ajoutée dans PHP 4.0.*

10.62.5 session_save_path

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [session_save_path](#)(string *path*)

[session_save_path\(\)](#) retourne le chemin du dossier utilisé pour enregistrer les données de sessions. Si *path* est fourni, le chemin prendra alors la valeur fournie. Note : *Sur certains systèmes d'exploitation, il vous faudra peut être fournir un chemin vers un système de sauvegarde qui peut gérer de grandes quantités de petits fichiers efficacement : par exemple, sous Linux, reiserfs peut être plus efficace que ext2fs.*

Note : *Cette fonction a été ajoutée dans PHP 4.0.*

10.62.6 session_id

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [session_id](#)(string *id*)

[session_id\(\)](#) retourne l'identifiant de session courante. Si *id* est fourni, il remplacera l'identifiant courant de la session.

La constante SID peut aussi être utilisée pour retrouver le nom de la session courante et son identifiant, comme chaîne à ajouter dans les URL. Note : *Cette fonction a été ajoutée dans PHP 4.0.*

10.62.7 session_register

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [session_register](#)(mixed *name*, mixed ...)

[session_register\(\)](#) enregistre une variable avec le nom *name* dans la session courante.

Cette fonction retourne true lorsque la variable est correctement enregistrée. Note : *Cette fonction a été ajoutée dans PHP 4.0.*

10.62.8 session_unregister

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [session_unregister](#)(string *name*)

[session_unregister\(\)](#) supprime la variable nommée *name* dans la session courante .

Cette fonction retourne true (vrai) lorsque la variable a été correctement supprimée de la session. Note : *Cette fonction a été ajoutée dans PHP 4.0.*

[10.62.9 session_is_registered](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [session_is_registered](#)(string *name*)

[session_is_registered\(\)](#) retourne true si il y a une variable du nom de *name* enregistrée dans la session courante. Note : *Cette fonction a été ajoutée dans PHP 4.0.*

[10.62.10 session_decode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [session_decode](#)(string *data*)

[session_decode\(\)](#) décode les données de session à partir de la chaîne *data*, et affecte les valeurs des variables de session. Note : *Cette fonction a été ajoutée dans PHP 4.0.*

[10.62.11 session_encode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [session_encode](#)

[session_encode\(\)](#) retourne les données de session dans une chaîne. Note : *Cette fonction a été ajoutée dans PHP 4.0.*

[10.63 Shared Memory Functions](#)

[\[Notes en ligne\]](#)

Shmop is an easy to use set of functions that allows php to read, write, create and delete UNIX shared memory segments. The functions will not work on windows, as it does not support shared memory. To use shmop you will need to compile php with the `--enable-shmop` parameter in your configure line.

Note : *The functions explained in the chapter begin all with @xref{function.shm- , , shm_() in PHP 4.0.3, but in PHP 4.0.4 and later versions these names are changed to begin with @xref{function.shmop- , , shmop_()}.*
} Shared Memory Operations Overview

```
<?php
// Create 100 byte shared memory block with system id if 0xff3
$shm_id = shmop_open(0xff3, "c", 0644, 100);
if(!$shm_id) {
```

```

echo "Couldn't create shared memory segment\n";
}
// Get shared memory block's size
$shm_size = shmop_size($shm_id);
echo "SHM Block Size: ".$shm_size. " has been created.\n";
// Lets write a test string into shared memory
$shm_bytes_written = shmop_write($shm_id, "my shared memory block", 0);
if($shm_bytes_written != strlen("my shared memory block")) {
echo "Couldn't write the entire length of data\n";
}
// Now lets read the string back
$my_string = shmop_read($shm_id, 0, $shm_size);
if(!$my_string) {
echo "Couldn't read from shared memory block\n";
}
echo "The data inside shared memory was: ".$my_string."\n";
//Now lets delete the block and close the shared memory segment
if(!shmop_delete($shm_id)) {
echo "Couldn't mark shared memory block for deletion.
}
shmop_close($shm_id);
?>

```

[10.63.1 shmop_open](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shmop_open](#)(int *key*, string *flags*, int *mode*, int *size*)

[shmop_open\(\)](#) can create or open a shared memory block.

[shmop_open\(\)](#) takes 4 parameters: key, which is the system's id for the shared memory block, this parameter can be passed as a decimal or hex. The second parameter are the flags that you can use:

- "a" for access (sets IPC_EXCL) use this flag when you need to open an existing shared memory segment
- "c" for create (sets IPC_CREATE) use this flag when you need to create a new shared memory segment.

The third parameter is the mode, which are the permissions that you wish to assign to your memory segment, those are the same as permission for a file. Permissions need to be passed in octal form ex. 0644. The last parameter is size of the shared memory block you wish to create in bytes. Note : *Note: the 3rd and 4th should be entered as 0 if you are opening an existing memory segment. On success [shmop_open\(\)](#) will return an id that you can use to access the shared memory segment you've created.*

Create a new shared memory block

```

<?php
$shm_id = shmop_open(0x0fff, "c", 0644, 100);
?>

```

This example opened a shared memory block with a system id of 0x0fff.

10.63.2 shmop_read

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [shmop_read](#) (int *shm*id, int *start*, int *count*)

[shmop_read\(\)](#) will read a string from shared memory block.

[shmop_read\(\)](#) takes 3 parameters: *shm*id, which is the shared memory block identifier created by [shmop_open\(\)](#), offset from which to start reading and count on the number of bytes to read.

Reading shared memory block

```
<?php
$shm_data = shmop_read($shm_id, 0, 50);
?>
```

This example will read 50 bytes from shared memory block and place the data inside \$shm_data.

10.63.3 shmop_write

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shmop_write](#) (int *shm*id, string *data*, int *offset*)

[shmop_write\(\)](#) will write a string into shared memory block.

[shmop_write\(\)](#) takes 3 parameters: *shm*id, which is the shared memory block identifier created by [shmop_open\(\)](#), *data*, a string that you want to write into shared memory block and offset, which specifies where to start writing data inside the shared memory segment.

Writing to shared memory block

```
<?php
$shm_bytes_written = shmop_write($shm_id, $my_string, 0);
?>
```

This example will write data inside \$my_string into shared memory block, \$shm_bytes_written will contain the number of bytes written.

10.63.4 shmop_size

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int @xref{function.shmop-size , , shmop_size (int *shm*id) }

@xref{function.shmop-size , , shmop_size()} is used to get the size, in bytes of the shared memory block. @xref{function.shmop-size , , shmop_size()} takes the *shm*id, which is the shared memory block identifier created by [shmop_open\(\)](#), the function will return and int, which represents the number of bytes the shared memory block occupies.

Getting the size of the shared memory block

```
<?php
$shm_size = shmop_size($shm_id);
?>
```

This example will put the size of shared memory block identified by \$shm_id into \$shm_size.

10.63.5 shmop_delete

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shmop_delete](#)(int *shm_id*)

[shmop_delete\(\)](#) is used to delete a shared memory block.

[shmop_delete\(\)](#) takes the shm_id, which is the shared memory block identifier created by [shmop_open\(\)](#). On success 1 is returned, on failure 0 is returned.

Deleting shared memory block

```
<?php
shmop_delete($shm_id);
?>
```

This example will delete shared memory block identified by \$shm_id.

10.63.6 shmop_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shmop_close](#)(int *shm_id*)

[shmop_close\(\)](#) is used to close a shared memory block.

[shmop_close\(\)](#) takes the shm_id, which is the shared memory block identifier created by [shmop_open\(\)](#).

Closing shared memory block

```
<?php
shmop_close($shm_id);
?>
```

This example will close shared memory block identified by \$shm_id.

10.64 SNMP functions

[\[Notes en ligne\]](#)

Afin de pouvoir utiliser les fonctions SNMP sous Unix, vous aurez besoin d'installer le package [UCD SNMP](#).

Sous Windows ces fonctions ne sont disponibles que sous NT, et pas sous Win95/98.

Important : Afin d'utiliser le package UCD SNMP, vous devez mettre la variable `NO_ZEROLENGTH_COMMUNITY` à 1 avant de compiler. Après avoir configuré UCD SNMP, éditez le fichier `config.h` et recherchez la valeur `NO_ZEROLENGTH_COMMUNITY`. Décommentez la ligne avec le `#define`. Cela doit ressembler à ceci :

```
#define NO_ZEROLENGTH_COMMUNITY 1
```

Si vous avez des erreurs "segmentation faults", lors de l'utilisation des commandes SNMP, c'est que vous n'avez pas suivi les recommandations précédentes. Si vous ne voulez pas recompiler UCD SNMP, vous pouvez aussi recompiler PHP avec l'option `--enable-ucd-snmp-hack` qui évitera cette erreur.

[10.64.1 snmpget](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [snmpget](#) (string *hostname*, string *community*, string *object_id*, int *timeout*, int *retries*)

Retourne un objet SNMP en cas de succès, et FALSE en cas d'erreur.

[snmpget\(\)](#) sert à lire une valeur d'un objet SNMP représenté par *object_id*. L'agent SNMP est défini par *hostname* et la communauté de lecture est spécifiée par le paramètre *community*.

```
$syscontact = snmpget("127.0.0.1", "public", "system.SysContact.0")
```

[10.64.2 snmpset](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [snmpset](#) (string *hostname*, string *community*, string *object_id*, string *type*, mixed *value*, int *timeout*, int *retries*)

Fixe la valeur de l'objet SNMP spécifié, en retournant TRUE en cas de succès et FALSE en cas d'erreur.

[snmpset\(\)](#) sert à affecter une valeur donnée à un objet SNMP, référencé par *object_id*. L'agent SNMP est défini par *hostname* et la communauté de lecture est spécifiée par le paramètre *community*.

[10.64.3 snmpwalk](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [snmpwalk](#) (string *hostname*, string *community*, string *object_id*, int *timeout*, int *retries*)

Retourne un tableau d'objets SNMP, en commençant à partir de *object_id* comme racine, ou FALSE en cas d'erreur.

[snmpwalk\(\)](#) sert à lire toutes les valeurs d'un agent SNMP, défini par *hostname*. *community* définit la communauté de lecture de l'agent. Un objet (*object_id* = null) sert de racine à l'arbre d'objet SNMP et tous les objets sous cette racine sont retournés dans un tableau. Si *object_id* est spécifié, tous les objets SNMP sous

cet objet sont retournés.

```
$a = snmpwalk("127.0.0.1", "public", "");
```

La fonction ci dessus va retourner tous les objets SNMP d'un agent SNMP qui fonctionnerait sur l'hôte local (localhost). Il suffit alors de faire une boucle pour travailler avec chacun des objets.

```
for ($i=0; $i<count($a); $i++) {
echo $a[$i];
}
```

10.64.4 snmpwalkoid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [snmpwalkoid](#)(string *hostname*, string *community*, string *object_id*, int *timeout* , int *retries*)

Retourne un tableau associatif, avec les identifiants d'objet et les objets associés, pour tous les objets situés sous la racine *object_id*, ou FALSE en cas d'erreur.

[snmpwalkoid\(\)](#) sert à lire tous les identifiants d'objet, et leur valeurs respectives, depuis un serveur SNMP. *community* indique la communauté de lecture pour cet agent. Un *object_id* null signifie qu'il faut utiliser la racine de l'arbre SNMP et tous les objets sous cet arbre seront retournés. Si *object_id* est spécifié, tous les objets SNMP situés sous cet objet seront retournés.

La fonction ci dessus va lire tous les objets de l'agent SNMP qui fonctionne sur l'hôte local. Il est alors possible de les passer en revue avec une boucle : l'existence de [snmpwalkoid\(\)](#) et [snmpwalk\(\)](#) est une question d'évolution. Ces deux fonctions sont fournies pour des raisons de compatibilité ascendante.

```
$a = snmpwalkoid("127.0.0.1", "public", "");
```

La fonction ci dessus va lire tous les objets de l'agent SNMP qui fonctionne sur l'hôte local. Il est alors possible de les passer en revue avec une boucle :

```
for (reset($a); $i = key($a); next($a)) {
echo "$i: $a[$i]<br>\n";
}
```

10.64.5 snmp_get_quick_print

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [snmp_get_quick_print](#)(void)

Retourne la valeur courante, stockée dans la librairie UCD, de l'option quick_print. Par défaut, quick_print est inactivée.

```
$quickprint = snmp_get_quick_print();
```

L'exemple ci-dessus devrait retourner FALSE, si quick_print est inactivée, et, l'exemple retournera TRUE quick_print est activée.

[snmp_get_quick_print\(\)](#) est seulement disponible avec la librairie UCD SNMP. Cette fonction n'est pas disponible avec la librairie Windows SNMP.

Voir : [snmp_set_quick_print\(\)](#) pour une description complète de l'affichage de quick_print.

10.64.6 snmp_set_quick_print

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [snmp_set_quick_print](#)(boolean *quick_print*)

Fixe la valeur de l'option quick_print de la librairie UCD SNMP. Lorsqu'elle a la valeur de (1), la librairie SNMP retournera des valeurs 'rapides'. Cela signifie que seule, la valeur sera retournée. Lorsqu'elle a la valeur de (0), la librairie va afficher d'autres informations (telles que l'adresse IP (IpAddress) ou OID). De plus, si quick_print n'est pas activée, la librairie affichera aussi des valeurs hexadécimales supplémentaires pour toutes les chaînes de trois caractères, ou moins.

Modifier quick_print est plus fréquent lorsqu'on utilise les valeurs retournées que lorsqu'on les affiche.

```
snmp_set_quick_print(0);
$a = snmpget("127.0.0.1", "public", ".1.3.6.1.2.1.2.2.1.9.1");
echo "$a<BR>\n";
snmp_set_quick_print(1);
$a = snmpget("127.0.0.1", "public", ".1.3.6.1.2.1.2.2.1.9.1");
echo "$a<BR>\n";
```

La première valeur affichée sera : 'Timeticks: (0) 0:00:00.00', tandis qu'avec quick_print activée, seul '0:00:00.00' sera affiché.

Par défaut, UCD SNMP retourne des valeurs détaillées, et quick_print sert à ne retourner que la valeur. Actuellement, les chaînes sont toujours retournées avec des guillemets supplémentaires. Ceci sera corrigé ultérieurement.

[snmp_set_quick_print\(\)](#) ne fonctionne qu'avec la librairie UCD SNMP. Cette fonction n'est pas disponible avec la librairie Windows SNMP.

10.65 Socket functions

[\[Notes en ligne\]](#)

The socket extension implements a low-level interface to the socket communication functions, providing the possibility to act as a socket server as well as a client.

For a more generic client-side socket interface, see [fsockopen\(\)](#) and [pfsockopen\(\)](#).

When using the socket functions described here, it is important to remember that while many of them have identical names to their C counterparts, they often have different declarations. Please be sure to read the descriptions to avoid confusion.

That said, those unfamiliar with socket programming can still find a lot of useful material in the appropriate Unix man pages, and there is a great deal of tutorial information on socket programming in C on the web, much of which can be applied, with slight modifications, to socket programming in PHP.

Socket example: Simple TCP/IP server

This example shows a simple talkback server. Change the **address** and **port** variables to suit your setup and execute. You may then connect to the server with a command similar to: telnet 192.168.1.53 10000 (where the address and port match your setup). Anything you type will then be output on the server side, and echoed back to you. To disconnect, enter 'quit'.

```
<?php
error_reporting(E_ALL);
/* Allow the script to hang around waiting for connections. */
set_time_limit(0);
$address = '192.168.1.53';
$port = 10000;
if (($sock = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
echo "socket() failed: reason: " . strerror($sock) . "\n";
}
if (($ret = bind($sock, $address, $port)) < 0) {
echo "bind() failed: reason: " . strerror($ret) . "\n";
}
if (($ret = listen($sock, 5)) < 0) {
echo "listen() failed: reason: " . strerror($ret) . "\n";
}
do {
if (($msgsock = accept_connect($sock)) < 0) {
echo "accept_connect() failed: reason: " . strerror($msgsock) . "\n";
break;
}
do {
$buf = '';
$ret = read($msgsock, $buf, 2048);
if ($ret < 0) {
echo "read() failed: reason: " . strerror($ret) . "\n";
break 2;
}
if ($ret == 0) {
break 2;
}
$buf = trim($buf);
if ($buf == 'quit') {
close($msgsock);
break 2;
}
$stalkback = "PHP: You said '$buf'.\n";
write($msgsock, $stalkback, strlen($stalkback));
echo "$buf\n";
} while (true);
close($msgsock);
} while (true);
close($sock);
?>
```

Socket example: Simple TCP/IP client

This example shows a simple, one-shot HTTP client. It simply connects to a page, submits a HEAD request, echoes the reply, and exits.

```

<?php
error_reporting(E_ALL);
echo "<h2>TCP/IP Connection</h2>\n";
/* Get the port for the WWW service. */
$service_port = getservbyname('www', 'tcp');
/* Get the IP address for the target host. */
$address = gethostbyname('www.php.net');
/* Create a TCP/IP socket. */
$socket = socket(AF_INET, SOCK_STREAM, 0);
if ($socket < 0) {
echo "socket() failed: reason: " . strerror($socket) . "\n";
} else {
    "socket() successful: " . strerror($socket) . "\n";
}
echo "Attempting to connect to '$address' on port '$service_port'...";
$result = connect($socket, $address, $service_port);
if ($result < 0) {
echo "connect() failed.\nReason: ($result) " . strerror($result) . "\n";
} else {
echo "OK.\n";
}
$in = "HEAD / HTTP/1.0\r\n\r\n";
$out = '';
echo "Sending HTTP HEAD request...";
write($socket, $in, strlen($in));
echo "OK.\n";
echo "Reading response:\n\n";
while (read($socket, $out, 2048)) {
echo $out;
}
echo "Closing socket...";
close($socket);
echo "OK.\n\n";
?>

```

[10.65.1 accept_connect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [accept_connect](#)(int *socket*)

After the socket *socket* has been created using [socket\(\)](#), bound to a name with [bind\(\)](#), and told to listen for connections with [listen\(\)](#), this function will accept incoming connections on that socket. Once a successful connection is made, a new socket descriptor is returned, which may be used for communication. If there are multiple connections queued on the socket, the first will be used. If there are no pending connections, [accept_connect\(\)](#) will block until a connection becomes present. If *socket* has been made non-blocking using [@xref{function.socket-set-blocking, , socket_set_blocking\(\)}](#) or [@xref{function.set-nonblock, , set_nonblock\(\)}](#), an error code will be returned.

The socket descriptor returned by [accept_connect\(\)](#) may not be used to accept new connections. The original listening socket *socket*, however, remains open and may be reused.

Returns a new socket descriptor on success, or a negative error code on failure. This code may be passed to [strerror\(\)](#) to get a textual explanation of the error.

See also [bind\(\)](#), [connect\(\)](#), [listen\(\)](#), [socket\(\)](#), and [strerror\(\)](#).

10.65.2 bind

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [bind](#)(int *socket*, string *address*, int *protocol*)

[bind\(\)](#) binds the name given in *address* to the socket described by *socket*, which must be a valid socket descriptor created with [socket\(\)](#).

The *address* parameter is either an IP address in dotted–quad notation (e.g. 127.0.0.1), if the socket is of the AF_INET family; or the pathname of a Unix–domain socket, if the socket family is AF_UNIX.

The *port* parameter is only used when connecting to an AF_INET socket, and designates the port on the remote host to which a connection should be made.

Returns zero on success, or a negative error code on failure. This code may be passed to [strerror\(\)](#) to get a textual explanation of the error.

See also [accept_connect\(\)](#), [connect\(\)](#), [listen\(\)](#), [socket\(\)](#), and [strerror\(\)](#).

10.65.3 connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [connect](#)(int *socket*, string *address*, int *port*)

Initiates a connection using the socket descriptor *socket*, which must be a valid socket descriptor created with [socket\(\)](#).

The *address* parameter is either an IP address in dotted–quad notation (e.g. 127.0.0.1), if the socket is of the AF_INET family; or the pathname of a Unix–domain socket, if the socket family is AF_UNIX.

The *port* parameter is only used when connecting to an AF_INET socket, and designates the port on the remote host to which a connection should be made.

Returns zero on success, or a negative error code on failure. This code may be passed to [strerror\(\)](#) to get a textual explanation of the error.

See also [bind\(\)](#), [listen\(\)](#), [socket\(\)](#), and [strerror\(\)](#).

10.65.4 listen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [listen](#)(int *socket*, int *backlog*)

After the socket *socket* has been created using [socket\(\)](#) and bound to a name with [bind\(\)](#), it may be told to listen for incoming connections on *socket*. A maximum of *backlog* incoming connections will be queued for processing.

[listen\(\)](#) is applicable only to sockets with type SOCK_STREAM or SOCK_SEQPACKET.

Returns zero on success, or a negative error code on failure. This code may be passed to [strerror\(\)](#) to get a textual explanation of the error.

See also [accept_connect\(\)](#), [bind\(\)](#), [connect\(\)](#), [socket\(\)](#), and [strerror\(\)](#).

10.65.5 socket

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [socket](#)(int *domain*, int *type*, int *protocol*)

Creates a communication endpoint (a socket), and returns a descriptor to the socket.

The *domain* parameter sets the domain. Currently, AF_INET and AF_UNIX are understood.

The *type* parameter selects the socket type. This is one of SOCK_STREAM, SOCK_DGRAM, SOCK_SEQPACKET, SOCK_RAW, SOCK_RDM, or SOCK_PACKET.

protocol sets the protocol.

Returns a valid socket descriptor on success, or a negative error code on failure. This code may be passed to [strerror\(\)](#) to get a textual explanation of the error.

For more information on the usage of [socket\(\)](#), as well as on the meanings of the various parameters, see the Unix man page ``socket (2)'`.

See also [accept_connect\(\)](#), [bind\(\)](#), [connect\(\)](#), [listen\(\)](#), and [strerror\(\)](#).

10.65.6 strerror

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strerror](#)(int *errno*)

[strerror\(\)](#) takes as its *errno* parameter the return value of one of the socket functions, and returns the corresponding explanatory text. This makes it a bit more pleasant to figure out why something didn't work; for instance, instead of having to track down a system include file to find out what '-111' means, you just pass it to [strerror\(\)](#), and it tells you what happened.

[strerror\(\)](#) *example*

```
<?php
if (($socket = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
echo "socket() failed: reason: " . strerror($socket) . "\n";
}
if (($ret = bind($socket, '127.0.0.1', 80)) < 0) {
echo "bind() failed: reason: " . strerror($ret) . "\n";
}
?>
```

The expected output from the above example (assuming the script is not run with root privileges):

```
bind() failed: reason: Permission denied
```

See also [accept_connect\(\)](#), [bind\(\)](#), [connect\(\)](#), [listen\(\)](#), and [socket\(\)](#).

10.66 Fonctions de chaîne de caractères

[\[Notes en ligne\]](#)

Ces fonctions permettent la manipulations de chaînes de caractères. Certaines sections plus spécialisées sont disponibles dès les sections sur les expressions régulières et dans la section URL.

[10.66.1 AddCSlashes](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [addslashes](#) (string *str*, string *charlist*)

Retourne une chaîne avec des backslash devant les caractères qui sont dans la liste *charlist*. Les caractères \n, \r etc. sont échappés. En langage C, les caractères avec un code ASCII inférieur à 32 ou supérieur à 126 sont convertis en représentation octale. Faites bien attention lorsque vous échappez des caractères alpha-numériques. Vous pouvez spécifier un intervalle dans *charlist* comme "\0..\37", qui échappera les caractères compris dans cet intervalle. *Exemple avec [addslashes\(\)](#)*

```
$escaped = addslashes ($no_echappe, "\0..\37!@\177..\377");
```

Note : Ajouté dans PHP4b3-dev.

Voir aussi [stripslashes\(\)](#), [stripslashes\(\)](#), [htmlspecialchars\(\)](#), [htmlspecialchars\(\)](#) et [quotemeta\(\)](#).

[10.66.2 AddSlashes](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [addslashes](#) (string *str*)

Retourne une chaîne avec des backslashes devant chaque caractère qui a en a besoin pour être inséré dans une requête de base de données. Ces caractères sont guillemets simples ('), guillemets doubles ("), backslash (\) et NULL (la valeur null).

Voir aussi [stripslashes\(\)](#), [htmlspecialchars\(\)](#) et [quotemeta\(\)](#).

[10.66.3 bin2hex](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bin2hex](#) (string *str*)

Retourne une chaîne ASCII contenant la représentation hexadécimale de *str*. La conversion est faite avec le bit de poids fort en premier.

[10.66.4 Chop](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [chop](#) (string *str*)

Retourne l'argument sans les espaces de fin de chaîne. *Exemple avec [chop\(\)](#)*

```
$trimmed = chop ($line);
```

Voir aussi [trim\(\)](#).

[10.66.5 Chr](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [chr](#)(int *ascii*)

Retourne le caractère de code ASCII *ascii*. *Exemple avec [chr\(\)](#)*

```
$str .= chr (27); /* ajoute un échappement à la fin de la chaîne $str */
/* Généralement, ceci est plus efficace */
$str = sprintf ("Cette chaîne se termine par un escape: %c", 27);
```

Cette fonction est le contraire de [ord\(\)](#). Voir aussi [sprintf\(\)](#) avec le format de chaîne %c.

[10.66.6 chunk_split](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [chunk_split](#)(string *string*, int *chunklen* , string *end*)

Permet de scinder une chaîne en plus petit morceaux, comme dans le cas de la conversion en [10.70.2 base64_encode](#) pour se conformer à la RFC 2045. Cette fonction insère une fin de chaîne *end* (par défaut "\r\n"), tous les *chunklen* (par défaut 76) caractères. La chaîne retournée est une nouvelle chaîne, et l'original n'est pas modifié. *Exemple avec [chunk_split\(\)](#)*

```
# formate $data avec la sémantique RFC 2045
$new_string = chunk_split (base64_encode($data));
```

Cette fonction est nettement plus rapide que [ereg_replace\(\)](#). Note : *Cette fonction a été ajoutée en 3.0.6.*

[10.66.7 convert_cyr_string](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [convert_cyr_string](#)(string *str*, string *from*, string *to*)

Cette fonction convertit la chaîne donnée depuis un alphabet cyrillique vers un autre. Les arguments *from* et *to* sont des caractères qui représentent la source et la destination. Les valeurs acceptées :

- k – koi8-r
- w – windows-1251
- i – iso8859-5
- a – x-cp866

- d – x–cp866
- m – x–mac–cyrillic

10.66.8 count_chars

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [count_chars](#)(string *string*, *mode*)

Compte le nombre d'occurrence de chaque octet (0..255) dans la chaîne *string* et le retourne de différente façon. L'option *Mode* prend, par défaut, la valeur 0. Suivant le *mode*, [count_chars\(\)](#) retourne une des réponses suivante :

- 0 – un tableau avec l'octet comme clé, et la fréquence comme valeur.
- 1 – Identique à 0, mais seule les fréquences non nulles sont listées.
- 2 – Identique à 0, mais seule les fréquences nulles sont listées.
- 3 – une chaîne qui contient tous les octets utilisés.
- 4 – une chaîne contenant tous les octets non utilisés.

Note : *Cette fonction a été ajoutée en PHP 4.0.*

10.66.9 crypt

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [crypt](#)(string *str*, string *salt*)

[crypt\(\)](#) va coder une chaîne en utilisant la méthode d'encryption du DES standard. Les arguments sont : la chaîne à encrypter, et un grain de sel qui servira de base pour l'encryption. Reportez vous au manuel Unix pour plus de détails.

Si le grain de sel n'est pas fourni, il sera automatiquement généré par PHP.

Certains systèmes d'exploitation acceptent plus d'un type d'encryption. En fait, le DES standard est parfois remplacé par une encryption MD5. Le type d'encryption est alors choisi en fonction du grain de sel. A l'installation, PHP détermine les possibilités de cryptage et décidera d'accepter d'autres grains de sel pour d'autres types d'encryption. Si le grain de sel n'est pas fourni, PHP générera alors un grain de 2 caractères, pour le DES standard, à moins que le système ne dispose de MD5 : dans ce cas, PHP générera un grain de sel pour MD, par défaut. PHP affecte la variable d'environnement CRYPT_SALT_LENGTH, à 2 si il utilise le DES standard, et à 12 si il utilise le MD5.

L'encryption standard fournit le grain de sel dans les deux premiers octets du résultat de la fonction [crypt\(\)](#).

Sur les systèmes qui supportent plusieurs méthodes d'encryption, les variables d'environnement suivantes sont mises à 0 ou à 1, en fonction de la disponibilité de la méthode :

- CRYPT_STD_DES – DES Standard avec 2–octets de SALT
- CRYPT_EXT_DES – DES étendu avec 9–octets SALT
- CRYPT_MD5 – MD5 avec 12–octets SALT commençant à \$1\$
- CRYPT_BLOWFISH – DES étendu avec 16–octets SALT commençant à \$2\$

Il n'y a pas d'algorithme de décryptage, étant donné que [crypt\(\)](#) est injective.

10.66.10 echo

[\[Notes en ligne\]](#) [\[Exemples\]](#)

echo (string *arg1*, string *argn...*)

Affiche tous les paramètres.

echo() n'est pas une fonction à proprement parler, ce qui rend l'usage des parenthèses facultatives. *Exemple avec echo()*

```
echo "Bonjour Monde";
echo "Cet echo se
répartis sur plusieurs lignes. Les nouvelles lignes
seront aussi affichées";
echo "et echo se\nrépartis sur plusieurs lignes. Les nouvelles lignes\nseront aussi affichées.";
```

Note : *En fait, si vous voulez passer plus d'un paramètre à echo(), vous ne DEVEZ pas les placer entre parenthèses.*

Voir aussi : [print\(\)](#), [printf\(\)](#) et [flush\(\)](#).

10.66.11 explode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array explode (string *separator*, string *string*)

Retourne un tableau qui contient les éléments de la chaîne, séparés par *separator*. *Exemple avec explode()*

```
$pizza = "piece1 piece2 piece3 piece4 piece5 piece6";
$pieces = explode (" ", $pizza);
```

Voir aussi [split\(\)](#) et [implode\(\)](#).

10.66.12 get_html_translation_table

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string get_html_translation_table (int *table*)

get_html_translation_table() retourne la table de traduction utilisée en interne par [htmlspecialchars\(\)](#) et [htmlentities\(\)](#). Il y a deux nouvelles définitions : (*HTML_ENTITIES*, *HTML_SPECIALCHARS*) qui vous permettent de spécifier vos propres tables. *Exemple de table de traduction*

```
$trans = get_html_translation_table (HTML_ENTITIES);
$str = "Hallo 62;Frau> Kr"mer";
$encoded = strtr ($str, $trans);
```

La variable \$encoded va contenir désormais : "Hallo ; Frau ; Kr>;mer".
[array_flip\(\)](#) est alors très efficace pour inverser la direction de traduction :

```
$strans = array_flip ($strans);
$original = strstr ($str, $strans);
```

Le contenu de \$original sera : "Hallo >Frau> & Kr”mer". Note : *Cette fonction a été ajoutée en PHP 4.0.*

Voir aussi : [htmlspecialchars\(\)](#), [htmlentities\(\)](#), [strstr\(\)](#), et [array_flip\(\)](#).

10.66.13 get_meta_tags

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [get_meta_tags](#) (string *filename*, int *use_include_path*)

Ouvre le fichier *filename* et l'analyse ligne par ligne, en recherchant les balises <meta>. *Meta Tags Example*

```
<meta name="author" content="name">
<meta name="tags" content="php3 documentation">
</head> <!-- parsing stops here -->
```

(Faites bien attention aux fins de lignes. PHP utilise une fonction native pour analyser le fichier d'entrée, ce qui fait que les fichiers faits sous Mac ne fonctionneront pas sous Unix).

Le nom d'une propriété devient sa clé, et la valeur devient la valeur dans le tableau associatif retourné, ce qui rend aisé la manipulation des informations. Les caractères spéciaux dans la nom de la propriété sont remplacés par des '_', le reste est converti en minuscule.

Mettre *use_include_path* à 1 forcera PHP à ouvrir les fichiers dans le chemin standard d'inclusion.

10.66.14 htmlentities

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [htmlentities](#) (string *string*)

Cette fonction est identique à [htmlspecialchars\(\)](#) en tous points, sauf que tous les caractères qui ont une entité équivalente en HTML sont remplacés par ces entités.

Actuellement, le jeu de caractères ISO-8859-1 character est utilisé.

Voir aussi [htmlspecialchars\(\)](#) et [nl2br\(\)](#).

10.66.15 htmlspecialchars

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [htmlspecialchars](#) (string *string*)

Certains caractères ont une valeur avec HTML, et doivent être remplacés par des balises HTML pour conserver leur valeur. Cette fonction retourne une chaîne avec tous les caractères sensibles remplacés par leur équivalent.

Cette fonction est utile pour empêcher un utilisateur de fournir un texte avec un sens HTML, comme dans un livre d'or.

Actuellement, PHP remplace les valeurs suivantes :

- '&' (et commercial) devient '&'
- '"' guillemet double) devient '"'
- '<' (inférieur à) devient '<'
- '>' (supérieur à) devient '>'

Notez bien que cette fonction ne fait aucun autre remplacement que ceux listés ci-dessus. Pour une traduction complète de toutes les balises, reportez vous à [htmlentities\(\)](#).

Voir aussi [htmlentities\(\)](#) et [nl2br\(\)](#).

[10.66.16 implode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [implode](#) (string *glue*, array *pieces*)

Retourne une chaîne constituée de tous les éléments du tableau, pris dans l'ordre, transformés en chaîne, et séparés par *glue*. *Exemple avec [implode\(\)](#)*

```
$colon_separated = implode (":", $array);
```

Voir aussi [explode\(\)](#), [join\(\)](#), et [split\(\)](#).

[10.66.17 join](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [join](#) (string *glue*, array *pieces*)

[join\(\)](#) [join\(\)](#) est un alias de [implode\(\)](#), et lui est identique en tous points.

Voir aussi [explode\(\)](#), [implode\(\)](#), et [split\(\)](#).

[10.66.18 ltrim](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ltrim](#) (string *str*)

Cette fonction enlève les caractères blancs placés au début d'une chaîne et retourne la chaîne raccourcie. Les caractères blancs sont : "\n", "\r", "\t", "\v", "\0", et " ".

Voir aussi [chop\(\)](#) et [trim\(\)](#).

[10.66.19 md5](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [md5](#) (string *str*)

Crypte la chaîne *str* en utilisant la méthode MD5 (voir [RSA Data Security. Inc. MD5 Message-Digest Algorithm.](#)).

10.66.20 Metaphone

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [metaphone](#)(string *str*)

Calcule la clé métaphone de la chaîne *str*.

Similairement à [soundex\(\)](#), métaphone crée une clé similaire pour des sons proches. C'est une fonction plus précise que [soundex\(\)](#) car elle prend en compte les règles basiques de la prononciation en anglais. Les clés métaphones sont de longueur variable.

Metaphone a été développé par Lawrence Philips <lphilips@verity.com>. Elle est décrit dans ["Practical Algorithms for Programmers", Binstock & Rex, Addison Wesley, 1995]. Note : *Cette fonction a été ajoutée en PHP 4.0.*

10.66.21 nl2br

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [nl2br](#)(string *string*)

Retourne la chaîne *string* dont toutes les lignes ont été remplacées par '
'.
Voir aussi [htmlspecialchars\(\)](#) et [htmlentities\(\)](#).

10.66.22 Ord

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ord](#)(string *string*)

Retourne la valeur ASCII du premier caractère de la chaîne *string*. Cette fonction est le contraire de [chr\(\)](#).
Exemple avec [ord\(\)](#)

```
if (ord ($str) == 10) {
echo "Le premier caractère de \"$str est un retour chariot.\n";
}
```

Voir aussi [chr\(\)](#).

10.66.23 parse_str

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [parse_str](#)(string *str*)

Analyse la chaîne *str* comme si c'était une chaîne passée par URL, et affecte les variables qu'elle y trouve.

Utilisation de [parse_str\(\)](#)

```
$str = "first=value&second[]=this+works&second[]=another";
parse_str($str);
echo $first;      /* prints "value" */
echo $second[0]; /* prints "this works" */
echo $second[1]; /* prints "another" */
```

10.66.24 [print](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

[_print](#) (string *arg*)

Affiche *arg*.

Voir aussi : [echo\(\)](#), [printf\(\)](#), et [flush\(\)](#).

10.66.25 [printf](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [printf](#) (string *format*, mixed *args*...)

Affiche les arguments en fonction du *format*. Ce format est décrit en détails dans la documentation de [sprintf\(\)](#).

Voir aussi : [print\(\)](#), [sprintf\(\)](#) et [flush\(\)](#).

10.66.26 [quoted_printable_decode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [quoted_printable_decode](#) (string *str*)

Cette fonction retourne une chaîne 8-bit résultant du décodage de la chaîne *str*. Cette fonction est similaire à [imap_qprint\(\)](#), hormis le fait qu'elle ne requiert pas le module IMAP.

10.66.27 [QuoteMeta](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [quotemeta](#) (string *str*)

Retourne une version de la chaîne *str*, avec un backslash (\) devant tous les caractères de la liste ci-dessous :

. \ + * ? [^] (\$)

Voir aussi [addslashes\(\)](#), [htmlentities\(\)](#), [htmlspecialchars\(\)](#), [nl2br\(\)](#) et [stripslashes\(\)](#).

10.66.28 rawurldecode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [rawurldecode](#)(string *str*)

Retourne une chaîne dont les séquences de caractères %xy, avec xy deux valeurs hexadécimales, auront été remplacées par le caractère ASCII correspondant. Par exemple, la chaîne @example foo%20bar%40baz devient @example foo bar@baz .

Voir aussi [rawurlencode\(\)](#).

10.66.29 rawurlencode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [rawurlencode](#)(string *str*)

Retourne une chaîne dont tous les caractères non-alpha-numériques (hormis @example _ .) auront été remplacés par des séquences %xy (%), avec xy deux valeurs hexadécimales. Ce codage est conforme à la RFC1738 qui évite que les caractères spéciaux soient interprétés comme des délimiteurs, et pour protéger les URL lors du transfert (contrairement à certains systèmes email). Par exemple, si vous voulez mettre un mot de passe dans une URL de ftp : **Exemple avec [rawurlencode\(\)](#) exemple 1**

```
echo '<A HREF="ftp://user:', rawurlencode ('foo @+%/'),
      '@ftp.my.com/x.txt">';
```

Ou, si vous transmettez un chemin dans une URL **Exemple avec [rawurlencode\(\)](#) exemple 2**

```
echo '<A HREF="http://x.com/department_list_script/',
      rawurlencode ('sales et marketing/Miami'), '">';
```

Voir aussi [rawurldecode\(\)](#).

10.66.30 setlocale

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [setlocale](#)(string *category*, string *locale*)

Category est une chaîne qui spécifie la catégorie de fonction qui va être affectée par les informations locales :

- LC_ALL Toutes les fonctions ci-dessous
- LC_COLLATE pour les comparaison de chaîne (en cours d'implémentation)
- LC_CTYPE pour la classification de caractères et les conversions, par exemple [strtoupper\(\)](#)
- LC_MONETARY pour localeconv() – (en cours d'implémentation)
- LC_NUMERIC pour les séparateurs décimaux
- LC_TIME pour le format des dates et heures date avec [strftime\(\)](#)

Si *locale* est une chaîne vide (""), les noms locaux prendront la valeur des variables d'environnement de même nom, ou à partir de "LANG".

Si *locale* vaut zéro ou "0", la valeur reste inchangée, mais l'état courant est retourné.

[setlocale\(\)](#) retourne la valeur courante, ou FALSE si la fonctionnalité n'est pas encore implémentée pour la plateforme. Une catégorie invalide provoque une alerte.

10.66.31 similar_text

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [similar_text](#)(string *first*, string *second*, double *percent*)

Cette fonction calcule la similarité entre deux chaînes, comme décrit par Oliver [1993]. Notez que cette implémentation n'utilise pas une pile, comme dans le pseudo-code d'Oliver, mais un appel récursif qui accélère parfois l'exécution. Notez aussi que la complexité de cet algorithme est en $O(N^{**3})$ avec N la taille de la plus grande chaîne.

En passant une référence comme troisième argument, [similar_text\(\)](#) va calculer le pourcentage de similarité. Il retourne le nombre de caractères correspondant l'un à l'autre, d'une chaîne à l'autre.

10.66.32 soundex

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [soundex](#)(string *str*)

Calcule la valeur soundex de *str*.

Une valeur Soundex est telle que deux mots prononcés de la même façon auront des valeurs Soundex identiques. Cela permet d'effectuer des recherches dans les bases de données, si vous connaissez la prononciation mais pas l'orthographe. Cette fonction retourne une chaîne de 4 caractères, commençant par une lettre.

Cette fonction particulière a été décrite par Donald Knuth dans "The Art Of Computer Programming, vol. 3: Sorting And Searching", Addison-Wesley (1973), pp. 391-392.

Exemple avec Soundex

```
soundex ("Euler") == soundex ("Ellery") == 'E460';
soundex ("Gauss") == soundex ("Ghosh") == 'G200';
soundex ("Knuth") == soundex ("Kant") == 'H416';
soundex ("Lloyd") == soundex ("Ladd") == 'L300';
soundex ("Lukasiewicz") == soundex ("Lissajous") == 'L222';
```

10.66.33 sprintf

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [sprintf](#)(string *format*, mixed *args*...)

Retourne une chaîne formatée avec le format *format*.

La chaîne de format est composée de 0 ou plus directives : généralement des caractères qui sont copiés tels

quel (hormis %), et des spécifications, chacune d'elle disposant de son propre paramètre. Cela s'applique à [sprintf\(\)](#) et [printf\(\)](#).

Chaque conversion consiste, dans l'ordre:

1. Une option de remplissage, qui indique quel caractère sera utilisé pour le remplissage, et la taille finale de la chaîne. Le caractère de remplissage peut être un espace ou le caractère zéro (0). La valeur par défaut est l'espace. Une autre valeur peut être spécifiée en la préfixant par un guillemet simple ('). Voir les exemples plus loin.
2. Un argument optionnel *alignment specifier* qui indique que le résultat doit être justifié à droite ou à gauche. Par défaut, il est justifié à gauche. Le caractère - signifie : justification à droite.
3. Argument optionnel, *width specifier* indique le nombre minimum de caractères que la conversion devrait retourner.
4. Argument optionnel, *precision specifier* indique le nombre de chiffres utilisé pour afficher un nombre à virgule flottante. Cette option n'a d'effet que sur les nombres à virgule, double. (Une autre fonction pratique pour formater les nombres est : [number_format\(\)](#).)
5. *type specifier* indique le type de données passées en argument : Les types possibles sont :
 - ◆ % – un signe pourcentage : aucun argument nécessaire.
 - ◆ b – l'argument est traité comme un entier, et représenté comme un nombre binaire.
 - ◆ c – l'argument est traité comme un entier, et représenté comme un nombre ascii.
 - ◆ d – l'argument est traité comme un entier, et représenté comme un nombre décimal.
 - ◆ f – l'argument est traité comme un double, et représenté comme un nombre à virgule flottante.
 - ◆ o – l'argument est traité comme un entier, et représenté comme un nombre octal.
 - ◆ s – l'argument est traité tel quel, et représenté comme une chaîne.
 - ◆ x – l'argument est traité comme un entier, et représenté comme un nombre hexadécimal (en minuscule).
 - ◆ X – l'argument est traité comme un entier, et représenté comme un nombre hexadécimal (en majuscule).

Voir aussi : [printf\(\)](#) et [number_format\(\)](#).

Exemple avec [sprintf\(\)](#): complété avec des zéros

```
$isodate = sprintf ("%04d-%02d-%02d", $year, $month, $day);
```

Exemple avec [sprintf\(\)](#): format monétaire

```
$money1 = 68.75;
$money2 = 54.35;
$money = $money1 + $money2;
// echo $money affichera "123.1";
$formatted = sprintf ("%01.2f", $money);
// echo $formatted affichera "123.10"
```

[10.66.34 strcasecmp](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strcasecmp](#) (string *str1*, string *str2*)

Retourne < 0 si *str1* est plus petit que *str2*; > 0 si *str1* est plus grand que *str2*, et 0 si ils sont égaux. *Exemple avec [strcasecmp\(\)](#)*

```
$var1 = "Bonjour";
$var2 = "bonjour";
if ( !strcasecmp($var1,$var2) ) {
echo '$var1 est égal à $var2, à la casse près.';
}
```

Voir aussi [ereg\(\)](#), [strcmp\(\)](#), [substr\(\)](#), [stristr\(\)](#), et [strstr\(\)](#).

[10.66.35 strchr](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strchr](#) (string *haystack*, string *needle*)

Cette fonction est un alias de [strstr\(\)](#), et lui est identique en tous points.

[10.66.36 strcmp](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strcmp](#) (string *str1*, string *str2*)

Retourne < 0 si *str1* est plus petit que *str2*; > 0 si *str1* est plus grand que *str2*, et 0 si ils sont égaux. Notez bien que la comparaison est sensible à la casse.

Voir aussi [ereg\(\)](#), [strcasecmp\(\)](#), [substr\(\)](#), [stristr\(\)](#) et [strstr\(\)](#).

[10.66.37 strcspn](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strcspn](#) (string *str1*, string *str2*)

Retourne la longueur du premier segment de la chaîne *str1* qui ne contiennent pas aucun des caractères de la chaîne *str2*.

Voir aussi [strspn\(\)](#).

[10.66.38 strip_tags](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strip_tags](#) (string *str*, string *allowable_tags*)

Cette fonction recherche et supprime toutes les balises HTML et PHP d'une chaîne. En cas de balises non fermées, ou de balises mal formées, elle génère une erreur. Cette fonction utilise le même système que la fonction [fgetss\(\)](#).

Vous pouvez utiliser l'option *allowable_tags* pour spécifier les balises qui seront ignorées. Note : *Allowable_tags* a été ajouté en PHP 3.0.13, et PHP4B3.

10.66.39 StripCSlashes

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [stripslashes](#)(string *str*)

Retourne une chaîne dont les backslashes ont été supprimés. Cette fonction reconnaît les \n, \r ..., et les représentations octales et hexadécimales utilisées en C. Note : *Ajouté dans PHP4b3-dev.*

Voir aussi [addslashes\(\)](#).

10.66.40 StripSlashes

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [stripslashes](#)(string *str*)

Retourne une chaîne dont tous les slashes ont été supprimés. (\' devient ', ... et ainsi de suite). Les doubles backslashes sont remplacés par des simples.

Voir aussi [addslashes\(\)](#).

10.66.41 striistr

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [striistr](#)(string *haystack*, string *needle*)

Retourne tous les éléments de *haystack* à partir de la première occurrence de *needle*, jusqu'à la fin. *needle* et *haystack* sont examinés sans tenir compte de la casse.

Si *needle* n'est pas trouvé, retourne FALSE.

Si *needle* n'est pas une chaîne, elle est convertie en entier, est utilisée comme si elle était passée à [chr\(\)](#).

Voir aussi [strchr\(\)](#), [strrchr\(\)](#), [substr\(\)](#), et [ereg\(\)](#).

10.66.42 strlen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strlen](#)(string *str*)

Retourne la longueur de la chaîne *string*.

10.66.43 strpos

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strpos](#) (string *haystack*, string *needle*, int *offset*)

Retourne la position numérique de la première occurrence de *needle* dans la chaîne *haystack*. Contrairement à [strrpos\(\)](#), *needle* peut être une chaîne.

Si *needle* n'est pas trouvée, retourne FALSE. Note : *Il est facile de confondre la valeur de retour "caractère trouvé à la position 0" et "caractère introuvable". Voici comment faire la différence :*

```
// PHP 4.0b3 et plus récent :
$pos = strpos ("b", $mystring);
if ($pos === false) { // note: trois égal signes
    // non trouvé

// versions plus anciennes que 4.0b3:
$pos = strpos ("b", $mystring);
if (is_string ($pos) 38; !$pos) {
    // non trouvé
}
}
```

Si *needle* n'est pas une chaîne, elle est convertie en entier, et utilisée comme la valeur ASCII d'un caractère.

L'argument optionnel *offset* permet de préciser le caractère à partir duquel chercher, dans *haystack*. La position doit être relative au début de la chaîne *haystack*.

Voir aussi [strrpos\(\)](#), [strrchr\(\)](#), [substr\(\)](#), [stristr\(\)](#) et [strstr\(\)](#).

10.66.44 strrchr

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strrchr](#) (string *haystack*, string *needle*)

Cette fonction retourne la partie de la chaîne *haystack* qui commence à la dernière occurrence de *needle* et va jusqu'à la fin de la chaîne *haystack*.

Retourne FALSE si *needle* n'est pas trouvé.

Si *needle* contient plus d'un caractère, les autres sont ignorés.

Si *needle* n'est pas une chaîne, il est converti en un entier, et utilisé comme valeur ordinale. *Exemple avec [strrchr\(\)](#)*

```
// lis le dernier repertoire de $PATH
$dir = substr (strrchr ($PATH, ":"), 1);
// lis tout après la dernière ligne
$text = "Line 1\nLine 2\nLine 3";
$last = substr (strrchr ($text, 10), 1 );
```

Voir aussi [substr\(\)](#), [stristr\(\)](#), et [strstr\(\)](#).

[10.66.45 str_repeat](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [str_repeat](#) (string *input*, int *multiplier*)

Retourne *input_str* répétées *multiplier* fois. *multiplier* doit être plus grand que 0.

Exemple avec [str_repeat\(\)](#)

```
echo str_repeat ("==", 10);
```

Cet exemple affichera "=====

Note : Cette fonction a été ajoutée en PHP 4.0.

[10.66.46 strrev](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strrev](#) (string *string*)

Retourne *string*, après avoir changé l'ordre des caractères.

[10.66.47 strrpos](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strrpos](#) (string *haystack*, char *needle*)

Retourne la position numérique de la dernière occurrence de *needle* dans la chaîne *haystack* string. Cette fonction ne peut accepter qu'un seul caractère.

Si *needle* n'est pas trouvé, retourne FALSE.

Si *needle* n'est pas une chaîne, elle est convertie en entier, et utilisée comme la valeur ASCII d'un caractère.

Voir aussi [strpos\(\)](#), [strrchr\(\)](#), [substr\(\)](#), [stristr\(\)](#) et [strstr\(\)](#).

[10.66.48 strspn](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strspn](#) (string *str1*, string *str2*)

Retourne la longueur du premier segment de *str1* qui est constitué entièrement de caractères dans la chaîne *str2*.

Voir aussi [strcspn\(\)](#).

[10.66.49 strstr](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strstr](#) (string *haystack*, string *needle*)

Retourne toute la chaîne *haystack* à partir de la première occurrence de *needle*, jusqu'à la fin.

Si *needle* n'est pas trouvé, retourne FALSE.

Si *needle* n'est pas une chaîne, elle est convertie en entier, et utilisée comme valeur ordinale d'un caractère.

Exemple avec [strstr\(\)](#)

```
$email = 'sterling@designmultimedia.com';
$domain = strstr ($email, '');
print $domain; // affiche designmultimedia.com
```

Voir aussi [strstr\(\)](#), [strchr\(\)](#), [substr\(\)](#), et [ereg\(\)](#).

10.66.50 strtok

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strtok](#) (string *arg1*, string *arg2*)

[strtok\(\)](#) est utilisée pour morceller une chaîne. Pour cela, si vous avez une chaîne du type "ceci est une chaîne exemple", vous pouvez la morceller en mots, en utilisant ' ' comme délimiteur. *Exemple avec [strtok\(\)](#)*

```
$string = "ceci est une chaîne exemple";
$tok = strtok ($string, " ");
while ($tok) {
echo "Mot=$tok<br>";
    $tok = strtok (" ");
}
```

Notez que seul, le premier appel à [strtok\(\)](#) utilise l'argument chaîne. Après, chaque appel à strtok ne requiert que le délimiteur à utiliser. Pour recommencer, vous pouvez simplement appeler [strtok\(\)](#) avec un nouvel argument, pour l'initialiser. Notez que vous pouvez mettre des délimiteurs multiples. La chaîne sera morcellée à chaque fois qu'on rencontrera un des délimiteurs.

Soyez prudents avec les délimiteurs qui sont égaux à "0". Cette valeur sera confondue avec FALSE.

Voir aussi [split\(\)](#) et [explode\(\)](#).

10.66.51 strtolower

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strtolower](#) (string *str*)

Retourne *string* avec tous les caractères alphabétiques en minuscule.

Notez que le caractère 'alphabétique' est déterminé par la table de caractères locale. Par exemple, dans la table des caractères par défaut du "C", des caractères tels que a-umlaut (") ne seront pas convertis.

Exemple avec [strtolower\(\)](#)

```
$str = "Marie A Un Petit Agneau, Et Elle L'Adore";
$str = strtolower($str);
print $str; # Affiche : marie a un petit agneau, et elle l'adore
```

Voir aussi [strtoupper\(\)](#) et [ucfirst\(\)](#).

[10.66.52 strtoupper](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strtoupper](#) (string *string*)

Retourne *string* avec tous ses caractères alphabétiques mis en majuscule.

Notez que le caractère 'alphabétique' est déterminé par la table de caractères locale. Par exemple, dans la table des caractères par défaut du "C", des caractères tels que a-umlaut (") ne seront pas convertis.

Exemple avec [strtoupper\(\)](#)

```
$str = "Marie A Un Petit Agneau, Et Elle L'Adore";
$str = strtoupper ($str);
print $str; # Affiche : MARIE A UN PETIT AGNEAU, ET ELLE L'ADORE
```

Voir aussi [strtolower\(\)](#) et [ucfirst\(\)](#).

[10.66.53 str_replace](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [str_replace](#) (string *needle*, string *str*, string *haystack*)

Cette fonction remplace toutes les occurrences de *needle* dans *haystack* par la chaîne *str*. Si vous n'avez pas besoin de règles de remplacement sophistiquées, vous pouvez toujours utiliser [ereg_replace\(\)](#).

Exemple avec [str_replace\(\)](#)

```
$bodytag = str_replace ("%body%", "black", "<body text=%body%>");
```

Cette fonction n'altère pas les données binaires.

Note : [str_replace\(\)](#) a été ajoutée dans PHP 3.0.6, mais était erronée jusqu'à PHP 3.0.8.

Voir aussi [ereg_replace\(\)](#) et [strtr\(\)](#).

[10.66.54 strtr](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strtr](#) (string *str*, string *from*, string *to*)

Cette fonction travaille sur *str*, remplaçant chaque occurrence de chaque caractère de la chaîne *from* correspondant à la chaîne *to* et retourne le résultat.

Si *from* et *to* sont de longueur différentes, les caractères en trop sont ignorés. *Exemple avec [strtr\(\)](#)*

```
$addr = strstr($addr, "AA-", "ao");
```

[strstr\(\)](#) peut aussi être appelée avec deux arguments. Dans ce cas, elle se comporte différemment : *from* doit être un tableau associatif contenant des paires de chaînes, qui seront remplacées dans la chaîne source.

[strstr\(\)](#) recherchera toujours la chaîne la plus longue, et la remplacera en premier. Elle ne remplacera jamais une portion de chaîne qu'elle a déjà remplacé.

Exemples:

```
$trans = array ("hello" => "hi", "hi" => "hello");
echo strstr("hi all, I said hello", $trans) . "\n";
```

Cete exemple affichera : "hello all, I said hi",

Note : *Travailler avec deux arguments a été ajouté dans PHP 4.0.*

Voir aussi [ereg_replace\(\)](#).

10.66.55 substr

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [substr](#) (string *string*, int *start*, int *length*)

[substr\(\)](#) retourne une portion de *string*, spécifiée avec le début *start* et la longueur *length*.

Si *start* est positif, la chaîne retournée commencera au caractère *start* de la chaîne *string*. Par exemple:

Exemples:

```
$rest = substr ("abcdef", 1); // retourne "bcdef"
$rest = substr ("abcdef", 1, 3); // retourne "bcd"
```

Si *start* est négatif, la chaîne retournée commencera au caractère *start* de la chaîne *string*, en partant de la fin.

Par exemple:

Exemples:

```
$rest = substr ("abcdef", -1); // retourne "f"
$rest = substr ("abcdef", -2); // retourne "ef"
$rest = substr ("abcdef", -3, 1); // retourne "d"
```

Si *length* est donné et positif, la chaîne retournée aura la longueur *length*. Si *length* est donné et négatif, la chaîne retournée aura la longueur *length*, en partant de la fin.

Exemples:

```
$rest = substr ("abcdef", 1, -1); // retourne "bcde"
```

Voir aussi [strchr\(\)](#) et [ereg\(\)](#).

[10.66.56 substr_replace](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [substr_replace](#) (string *string*, string *replacement*, int *start*, int *length*)

[substr_replace\(\)](#) effectue un remplacement dans la portion de *string* délimitée par le caractère *start* et de longueur optionnelle *length*. Le remplacement est fait avec la chaîne *replacement*. Le résultat est retourné. Si *start* est positif, le remplacement commencera au caractère *start*, dans la chaîne *string*. Si *start* est négative, le remplacement commencera au caractère *start* en partant de la fin de la chaîne *string*. Si *length* est donné et positif, la chaîne retournée aura la longueur *length*. Si *length* est donné et négatif, la chaîne retournée aura la longueur *length*, en partant de la fin.

Exemple avec [substr_replace\(\)](#)

```
<?php
$var = 'ABCDEFGH:/MNRPQR/';
echo "Original: $var<hr>\n";
/* Ces deux exemples remplacent tout $var avec 'bob'. */
echo substr_replace ($var, 'bob', 0) . "<br>\n";
echo substr_replace ($var, 'bob', 0, strlen ($var)) . "<br>\n";
/* Insère 'bob' à gauche, du début de $var. */
echo substr_replace ($var, 'bob', 0, 0) . "<br>\n";
/* Ces deux exemples remplacent 'MNRPQR' dans $var avec 'bob'. */
echo substr_replace ($var, 'bob', 10, -1) . "<br>\n";
echo substr_replace ($var, 'bob', -7, -1) . "<br>\n";
/* Efface 'MNRPQR' dans $var. */
echo substr_replace ($var, '', 10, -1) . "<br>\n";
?>
```

Voir aussi [str_replace\(\)](#) et [substr\(\)](#).

Note : [substr_replace\(\)](#) a été ajoutée dans PHP 4.0.

[10.66.57 trim](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [trim](#) (string *str*)

Cette fonction retire les espaces blancs de début et de fin de chaîne, et retourne la chaîne nettoyée. Les espaces blancs sont : "\n", "\r", "\t", "\v", "\0", et " " (espace).

Voir aussi [chop\(\)](#) et [ltrim\(\)](#).

[10.66.58 ucfirst](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ucfirst](#) (string *str*)

Met le premier caractère d'une chaîne *str* en majuscule, si ce caractère est alphabétique.

Notez que le caractère 'alphabétique' est déterminé par la table de caractères locale. Par exemple, dans la table des caractères par défaut du "C", des caractères tels que a-umlaut (") ne seront pas convertis. **Exemple avec**

[ucfirst\(\)](#)

```
$text = 'marie a un petit agneau, et l'adore.';
$text = ucfirst($text); // $text vaut : Marie a un petit agneau, et l'adore
```

Voir aussi [strtoupper\(\)](#) et [strtolower\(\)](#).

[10.66.59 ucwords](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ucwords\(\)](#)(string *str*)

Met le premier caractère de chaque mot de la chaîne *str* si ce caractère est une lettre. *Exemple avec [ucwords\(\)](#)*

```
$text = "marie a un petit agneau, et l'adore.";
$text = ucwords($text); // $text vaut : Marie A Un Petit Agneau, Et l'Adore.
```

Voir aussi [strtoupper\(\)](#), [strtolower\(\)](#) et [ucfirst\(\)](#).

[10.67 Fonctions Shockwave Flash](#)

[\[Notes en ligne\]](#)

PHP a la capacité de créer des animations Shockwave Flash gr'ce au module de Paul Haerberli : libswf module. Vous pouvez télécharger libswf à <http://reality.sgi.com/grafica/flash/>. Une fois que vous avez libswf, tout ce qui reste à faire est de configurer PHP avec `--with-swf[=DIR]` où DIR est le dossier qui accueille les dossiers de include et lib. Le dossier include doit contenir le fichier swf.h file et le dossier lib doit contenir le fichier libswf.a. Si vous décompressez la distribution de libswf, les deux fichiers seront dans le même dossier. Par conséquent, vous devrez les mettre dans le dossier ad hoc manuellement.

Une fois que vous avez réussi à installer PHP avec Shockwave Flash, vous pouvez créer des animations Flash avec PHP. Vous serez surpris du résultat. Essayez donc ceci : *Exemple SWF*

```
<?php
swf_openfile ("test.swf", 256, 256, 30, 1, 1, 1);
swf_orho2 (-100, 100, -100, 100);
swf_defineline (1, -70, 0, 70, 0, .2);
swf_definerect (4, 60, -10, 70, 0, 0);
swf_definerect (5, -60, 0, -70, 10, 0);
swf_addcolor (0, 0, 0, 0);
swf_definefont (10, "Mod");
swf_fontsize (5);
swf_fontslant (10);
swf_definetext (11, "Voici le mariage de FLASH et PHP!", 1);
swf_pushmatrix ();
swf_translate (-50, 80, 0);
swf_placeobject (11, 60);
```

```

swf_popmatrix ();
for ($i = 0; $i < 30; $i++) {
    $p = $i/(30-1);
    swf_pushmatrix ();
    swf_scale (1-($p*.9), 1, 1);
    swf_rotate (60*$p, 'z');
    swf_translate (20+20*$p, $p/1.5, 0);
    swf_rotate (270*$p, 'z');
    swf_addcolor ($p, 0, $p/1.2, -$p);
    swf_placeobject (1, 50);
    swf_placeobject (4, 50);
    swf_placeobject (5, 50);
    swf_popmatrix ();
    swf_showframe ();
}
for ($i = 0; $i < 30; $i++) {
    swf_removeobject (50);
    if (($i%4) == 0) {
        swf_showframe ();
    }
}
swf_startdoaction ();
swf_actionstop ();
swf_enddoaction ();
swf_closefile ();
?>

```

Cela va produire une animation, proche de [celle ci](#) (mais traduite en anglais).

Note : *Le support de Flash a été ajouté dans PHP4 RC2.*

[10.67.1 swf_openfile](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_openfile](#)(string *filename* , float *width* , float *height* , float *framerate* , float *r* , float *g* , float *b*)

[swf_openfile\(\)](#) crée un nouveau fichier *filename* de largeur *width*, et de hauteur *height*, à la vitesse de *framerate*, de couleur de fond RGB (*r*, *g*, *b*).

[swf_openfile\(\)](#) doit être la première fonction à appeler, sous peine d'erreur mémoire (segmentation fault). Si vous voulez envoyer votre production au client HTML, utilisez le nom de fichier "php://stdout" (le support de ceci est prévue pour la version 4.0.1 et ultérieur).

[10.67.2 swf_closefile](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_closefile](#)

Ferme le fichier courant, qui a été ouvert avec [swf_openfile\(\)](#).

[10.67.3 swf_labelframe](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_labelframe](#)(string *name*)

Donne le nom *name* au frame courant.

[10.67.4 swf_showframe](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_showframe](#)

[swf_showframe\(\)](#) affiche le frame courant.

[10.67.5 swf_setframe](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_setframe](#) (int *framenumber*)

[swf_setframe\(\)](#) selectionne le frame *framenumber* comme frame actif.

[10.67.6 swf_getframe](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [swf_getframe](#)

[swf_getframe\(\)](#) retourne le numéro de frame courant.

[10.67.7 swf_mulcolor](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_mulcolor](#) (float *r* , float *g* , float *b* , float *a*)

[swf_mulcolor\(\)](#) fixe la valeur globale de multiplication (the global multiply color...) à la couleur *rgba*. Cette couleur est utilisée (implicitement) par [swf_placeobject\(\)](#), [swf_modifyobject\(\)](#) et [swf_addbuttonrecord\(\)](#). La couleur d'un objet sera multipliée par *rgba* lorsque l'objet est placé sur la scène.

Note : *Les valeurs de rgba peuvent être positives ou négatives.*

[10.67.8 swf_addcolor](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_addcolor](#) (float *r* , float *g* , float *b* , float *a*)

[swf_addcolor\(\)](#) fixe la valeur globale de multiplication (the global multiply color...) à la couleur *rgba*. Cette couleur est utilisée (implicitement) par [swf_placeobject\(\)](#), [swf_modifyobject\(\)](#) et [swf_addbuttonrecord\(\)](#). La couleur d'un objet sera ajouté à *rgba* lorsque l'objet est placé sur la scène.

Note : *Les valeurs de rgba peuvent être positives ou négatives.*

[10.67.9 swf_placeobject](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_placeobject](#)(int *objid* , int *depth*)

Place l'objet *objid* dans le frame courant, à la profondeur *depth*. *objid* et *depth* doivent être compris entre 1 et 65535.

Cette fonction utilise la couleur courante de multiplication (spécifiée par [swf_mulcolor\(\)](#)) et la couleur courante d'addition (spécifiée par [swf_addcolor\(\)](#)) pour colorer l'objet, et utilise la matrice courante pour positionner l'objet.

Note : *Le support des couleurs RGBA est complet.*

[10.67.10 swf_modifyobject](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_modifyobject](#)(int *depth* , int *how*)

Modifie la position et/ou la couleur de l'objet situé à la profondeur de *depth*. L'argument *how* détermine ce qui doit être modifié. *how* peut prendre les valeurs de MOD_MATRIX, MOD_COLOR ou la combinaison des deux.

MOD_COLOR utilise la couleur courante de multiplication (spécifiée par [swf_mulcolor\(\)](#)) et la couleur courante d'addition (spécifiée par [swf_addcolor\(\)](#)) pour colorer l'objet, et MOD_MATRIX utilise la matrice courante pour positionner l'objet.

[10.67.11 swf_removeobject](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_removeobject](#)(int *depth*)

Enlève l'objet situé à la profondeur *depth* de la scène.

[10.67.12 swf_nextid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [swf_nextid](#)

[swf_nextid\(\)](#) retourne le prochain identifiant d'objet libre.

[10.67.13 swf_startdoaction](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_startdoaction](#)

[swf_startdoaction\(\)](#) commence la description d'une liste d'action pour la frame courante. Cette fonction doit être appelée avant que les actions ne soient définies pour le cadre courant.

[10.67.14 swf_actiongotoframe](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actiongotoframe](#)(int *framenumbers*)

[swf_actiongotoframe\(\)](#) se déplace jusqu'au frame *framenumber*, le joue, puis s'arrête.

10.67.15 swf_actiongeturl

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actiongeturl](#) (string *url* , string *target*)

[swf_actiongeturl\(\)](#) lit l'URL *url*, avec la destination *target*.

10.67.16 swf_actionnextframe

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actionnextframe](#)

[swf_actionnextframe\(\)](#) avance d'un frame le frame courant.

10.67.17 swf_actionprevframe

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actionprevframe](#)

[swf_actionprevframe\(\)](#) recule d'un frame le frame courant.

10.67.18 swf_actionplay

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actionplay](#)

[swf_actionplay\(\)](#) joue l'animation flash à partir du frame courant.

10.67.19 swf_actionstop

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actionstop](#)

Arrête l'animation flash au frame courant.

10.67.20 swf_actiontogglequality

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actiontogglequality](#)

Choisi le niveau de qualité haut ou bas.

[10.67.21 swf_actionwaitforframe](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actionwaitforframe](#) (int *framenumbers* , int *skipcount*)

[swf_actionwaitforframe\(\)](#) vérifie que le frame *framenumbers* a bien été chargé. Si ce n'est pas le cas, elle ignore les actions *skipcount*. Cela est très utile pour les séquences du type "Chargement...".

[10.67.22 swf_actionsettarget](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actionsettarget](#) (string *target*)

[swf_actionsettarget\(\)](#) fixe le contexte des actions. Vous pouvez utiliser cette fonction pour contrôler d'autres animations Flash qui seraient en fonctionnement.

[10.67.23 swf_actiongotolabel](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actiongotolabel](#) (string *label*)

[swf_actiongotolabel\(\)](#) affiche le frame de nom *label*, puis stoppe.

[10.67.24 swf_enddoaction](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_enddoaction](#)

[swf_startdoaction\(\)](#) termine l'action courante.

[10.67.25 swf_defineline](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_defineline](#) (int *objid* , float *x1* , float *y1* , float *x2* , float *y2* , float *width*)

[swf_defineline\(\)](#) définit une ligne commençant aux coordonnées (*x1*, *y1*), et finissant au point de coordonnées (*x2*, *y2*). Elle aura la largeur de *width*.

[10.67.26 swf_definerect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_definerect](#) (int *objid* , float *x1* , float *y1* , float *x2* , float *y2* , float *width*)

[swf_definerect\(\)](#) définit un rectangle, de coin supérieur gauche aux coordonnées (*x1*, *y1*), et de coin inférieur droit aux coordonnées (*x2*, *y2*). L'épaisseur des bords est donnée par le paramètre *width*. *width*, 0.0 le rectangle sera rempli.

[10.67.27 swf_definepoly](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_definepoly](#) (int *objid* , array *coords* , int *npoints* , float *width*)

[swf_definepoly\(\)](#) définit un polygone, dont les coordonnées des sommets sont placés dans le tableau *coords*). *npoints* est le nombre de points contenu dans le tableau *coords*. *width* est la largeur des bords du polygone. Si *width* vaut 0.0, le polygone sera rempli.

[10.67.28 swf_startshape](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_startshape](#) (int *objid*)

[swf_startshape\(\)](#) commence une forme complexe, qui sera reperé par l'identifiant d'objet *objid*.

[10.67.29 swf_shapelinesolid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapelinesolid](#) (float *r* , float *g* , float *b* , float *a* , float *width*)

[swf_shapelinesolid\(\)](#) permet de choisir le style de ligne, à savoir la couleur et la largeur. Si *width* vaut 0.0, les lignes ne seront pas dessinées.

[10.67.30 swf_shapefilloff](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapefilloff](#)

[swf_shapefilloff\(\)](#) inactive le remplissage pour la forme courante.

[10.67.31 swf_shapefillsolid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapefillsolid](#) (float *r* , float *g* , float *b* , float *a*)

[swf_shapefillsolid\(\)](#) fixe la couleur pour le style courant de remplissage à *rgba*.

[10.67.32 swf_shapefillbitmapclip](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapefillbitmapclip](#) (int *bitmapid*)

Choisi le mode de remplissage par texture : les espaces vides seront remplis avec la bitmap *bitmapid*.

[10.67.33 swf_shapefillbitmaptile](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapefillbitmaptile](#)(int *bitmapid*)

Choisi le mode de remplissage par texture : les espaces vides seront remplis avec la bitmap *bitmapid*, répétée autant de fois qu'il le faut.

[10.67.34 swf_shapemoveto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapemoveto](#)(float *x* , float *y*)

[swf_shapemoveto\(\)](#) fixe la position courante au point de coordonnées (*x*, *y*).

[10.67.35 swf_shapelineto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapelineto](#)(float *x* , float *y*)

[swf_shapelineto\(\)](#) dessine une ligne entre la position courante et le points de coordonnées (*x*, *y*). La position courante devient alors (*x*, *y*).

[10.67.36 swf_shapecurveto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapecurveto](#)(float *x1* , float *y1* , float *x2* , float *y2*)

[swf_shapecurveto\(\)](#) dessine la courbe de Bézier quadratique entre les points de coordonnées (*x1* , *y1*) et (*x2*, *y2*). La position courante devient alors (*x2*, et *y2*).

[10.67.37 swf_shapecurveto3](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapecurveto3](#)(float *x1* , float *y1* , float *x2* , float *y2* , float *x3* , float *y3*)

Dessine une courbe de Bézier cubique, en utilisant les points de coordonnées (*x1*, *y1*) et (*x2*,*y2*) comme points de contrôle, et le point de coordonnées (*x3*, *y3*) comme point final. La position finale devient alors la position courante.

[10.67.38 swf_shapearc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapearc](#)(float *x* , float *y* , float *r* , float *ang1* , float *ang2*)

[swf_shapearc\(\)](#) dessine un arc de cercle, depuis l'angle *ang1* jusqu'à l'angle *ang2*. Le centre du cercle est aux

coordonnées (x, y) , et de rayon r .

[10.67.39 swf_endshape](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_endshape](#)

[swf_endshape\(\)](#) complète la définition de la forme courante.

[10.67.40 swf_definefont](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_definefont](#)(int *fontid* , string *fontname*)

[swf_definefont\(\)](#) définit la police *fontname* et lui affecte l'identifiant *fontid*. Cette police devient alors la police courante.

[10.67.41 swf_setfont](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_setfont](#)(int *fontid*)

[swf_setfont\(\)](#) remplace la police courante par la police repérée par l'identifiant *fontid*.

[10.67.42 swf_fontsize](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_fontsize](#)(float *size*)

[swf_fontsize\(\)](#) remplace la taille de la police par la taille *size*.

[10.67.43 swf_fontslant](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_fontslant](#)(float *slant*)

[swf_fontslant\(\)](#) fixe l'inclinaison de la police courante à *slant*. Les valeurs positives values créeront une inclinaison vers la droite, et les valeurs négatives, vers la gauche.

[10.67.44 swf_fontracking](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_fontracking](#)(float *tracking*)

[swf_fontracking\(\)](#) change l'espacement, et lui affecte la valeur de *tracking*. Cette fonction sert à accroître l'espace entre les lettres et le texte. Les valeurs positives accroissent cet espace, et les valeurs négatives le

réduisent.

[10.67.45 swf_getfontinfo](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [swf_getfontinfo](#)

[swf_getfontinfo\(\)](#) retourne la hauteur du A majuscule, et du x minuscule, dans un tableau associatif :

- Aheight – La hauteur du A majuscule, en pixels.
- xheight – La hauteur du x minuscule, en pixels.

[10.67.46 swf_definetext](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_definetext](#) (int *objid* , string *str* , int *docenter*)

Définit la chaîne de texte *str*, en utilisant la police courante. *docenter* indique si la chaîne doit être centrée (valeur de 1), ou pas.

[10.67.47 swf_textwidth](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [swf_textwidth](#) (string *str*)

[swf_textwidth\(\)](#) retourne la longueur de la chaîne *str*, en pixels, en utilisant la police courante.

[10.67.48 swf_definebitmap](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_definebitmap](#) (int *objid* , string *image_name*)

The [swf_definebitmap\(\)](#) function defines a bitmap given a GIF, JPEG, RGB or FI image. The image will be converted into a Flash JPEG or Flash color map format.

[10.67.49 swf_getbitmapinfo](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [swf_getbitmapinfo](#) (int *bitmapid*)

[swf_getbitmapinfo\(\)](#) retourne un tableau d'information sur l'image bitmap repérée par *bitmapid*. Le tableau a les éléments suivant :

- "size" – La taille en octets de l'image.
- "width" – La largeur en pixels de l'image.

- "height" – La hauteur en pixels de l'image.

[10.67.50 swf_startsymbol](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_startsymbol](#)(int *objid*)

Définit un identifiant d'objet comme symbole. Les symboles sont des petites animations flash qui peuvent être jouées simultanément. *objid* est l'identifiant d'objet que vous voulez définir comme symbole.

[10.67.51 swf_endsymbol](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_endsymbol](#)

[swf_endsymbol\(\)](#) termine la définition de symbole, qui a été commencée avec [swf_startsymbol\(\)](#).

[10.67.52 swf_startbutton](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_startbutton](#)(int *objid* , int *type*)

[swf_startbutton\(\)](#) commence la définition d'un bouton. *type* peut prendre les valeurs de TYPE_MENUBUTTON ou TYPE_PUSHBUTTON. La constante TYPE_MENUBUTTON permet au focus de traverser lorsque la souris est cliquée, alors que TYPE_PUSHBUTTON ne le permet pas.

[10.67.53 swf_addbuttonrecord](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_addbuttonrecord](#)(int *states* , int *shapeid* , int *depth*)

[swf_addbuttonrecord\(\)](#) permet de modifier les caractéristiques d'un bouton. *states*, définit les états du bouton autorisés : ce peut être : BSHitTest, BSDown, BSOver ou BSUp. *shapeid* est l'apparence du bouton, c'est à dire l'objet qui représente le bouton. *depth* est la profondeur de placement du bouton, dans le frame courant. exemple avec [swf_addbuttonrecord\(\)](#) *exemple avec [swf_addbuttonrecord\(\)](#)*

```
swf_startButton ($objid, TYPE_MENUBUTTON);
swf_addButtonRecord (BSDown|BSOver, $buttonImageId, 340);
swf_onCondition (MenuEnter);
swf_actionGetUrl ("http://www.designmultimedia.com", "_level1");
swf_onCondition (MenuExit);
swf_actionGetUrl ("", "_level1");
swf_endButton ();
```

10.67.54 swf_oncondition

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_oncondition](#) (int *transition*)

[swf_oncondition\(\)](#) décrit une transition qui va déclencher une liste d'actions. Il y a plusieurs types de transitions possibles, les suivantes sont destinées aux boutons de type TYPE_MENUBUTTON:

- IdletoOverUp
- OverUptoIdle
- OverUptoOverDown
- OverDowntoOverUp
- IdletoOverDown
- OutDowntoIdle
- MenuEnter (IdletoOverUp|IdletoOverDown)
- MenuExit (OverUptoIdle|OverDowntoIdle)

Pour les types TYPE_PUSHBUTTON voici les options :

- IdletoOverUp
- OverUptoIdle
- OverUptoOverDown
- OverDowntoOverUp
- OverDowntoOutDown
- OutDowntoOverDown
- OutDowntoIdle
- ButtonEnter (IdletoOverUp|OutDowntoOverDown)
- ButtonExit (OverUptoIdle|OverDowntoOutDown)

10.67.55 swf_endbutton

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_endbutton](#)

[swf_endbutton\(\)](#) termine la définition du bouton courant.

10.67.56 swf_viewport

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_viewport](#) (double *xmin* , double *xmax* , double *ymin* , double *ymax*)

[swf_viewport\(\)](#) sélectionne une nouvelle zone pour y dessiner ultérieurement. La zone est définie de *xmin* à *xmax* et de *ymin* à *ymax*. Si cette fonction n'est pas appelée, les valeurs par défaut sont celles de l'écran courant.

10.67.57 swf_ortho

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_ortho](#)(double *xmin* , double *xmax* , double *ymin* , double *ymax* , double *zmin* , double *zmax*)

[swf_ortho\(\)](#) définit une projection orthogonale entre les coordonnées utilisateur et le port courant.

10.67.58 swf_ortho2

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_ortho2](#)(double *xmin* , double *xmax* , double *ymin* , double *ymax*)

[swf_ortho2\(\)](#) définit une projection orthogonale à 2 dimensions entre les coordonnées utilisateur et le port courant. C'est la projection par défaut des animations Flash. Si vous souhaitez une perspective, utilisez plutôt [swf_perspective\(\)](#).

10.67.59 swf_perspective

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_perspective](#)(double *fovy* , double *aspect* , double *near* , double *far*)

[swf_perspective\(\)](#) définit une projection orthogonale à 3 dimensions entre les coordonnées utilisateur et le port courant. Le paramètre *fovy* est l'angle de vue de la direction y. Le paramètre *aspect* doit être choisi pour correspondre au ratio de la vue utilisée. *near* est le plan adjacent proche *far* est le plan adjacent distant. Note : *Diverses distortions peuvent apparaître lors de ce genre de projection, car Flash ne dispose que d'une matrice à 2 dimensions. Certaines distortions font vraiment t'che d'encre.*

10.67.60 swf_polarview

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_polarview](#)(double *dist* , double *azimuth* , double *incidence* , double *twist*)

[swf_polarview\(\)](#) définit la position de l'utilisateur en coordonnées polaires. *dist* est la distance entre le point de vue et l'origine. *azimuth* définit l'angle azimutal dans le plan x,y mesuré en distance depuis l'axe y. *incidence* définit l'angle d'incidence dans le plan y,z, mesuré en distance depuis l'axe z. Finalement, *twist* est l'angle de rotation du point de vue sur la ligne de vue, en utilisant la règle de la main droite.

10.67.61 swf_lookat

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_lookat](#)(double *view_x* , double *view_y* , double *view_z* , double *reference_x* , double *reference_y* , double *reference_z* , double *twist*)

[swf_lookat\(\)](#) définit une transformation de vue, en donnant la position de la vue, de coordonnées (*view_x*, *view_y*, et *view_z*) et les coordonnées du point de référence dans la scène, de coordonnées (*reference_x*,

reference_y , *reference_z*). Le paramètre *twist* contrôle la rotation le long de l'axe des z de l'utilisateur.

[10.67.62 swf_pushmatrix](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_pushmatrix](#)

[swf_pushmatrix\(\)](#) empile la matrice de transformation courante dans la pile.

[10.67.63 swf_popmatrix](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_popmatrix](#)

[swf_popmatrix\(\)](#) dépile la matrice de transformation.

[10.67.64 swf_scale](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_scale](#)(double *x* , double *y* , double *z*)

[swf_scale\(\)](#) fait une mise à l'échelle de *x* pour les coordonnées x, de *y* pour les coordonnées y et *z* pour les coordonnées z.

[10.67.65 swf_translate](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_translate](#)(double *x* , double *y* , double *z*)

[swf_translate\(\)](#) translate la transformation courante de *x*, *y*, et *z*, dans les directions x, y et z.

[10.67.66 swf_rotate](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_rotate](#)(double *angle* , string *axis*)

[swf_rotate\(\)](#) fait subir la rotation d'angle *angle*, autour de l'axe *axis*. Les valeurs possibles pour *axis* sont : 'x' (axe x), 'y' (axe y) ou 'z' (axe z).

[10.67.67 swf_posround](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_posround](#)(int *round*)

[swf_posround\(\)](#) active ou désactive l'approximation lors des translations, lorsque des objets sont placés ou déplacés. Il y a des situations où le texte devient plus lisible lorsque l'approximation a été activée.

round active l'approximation (1) ou la désactive (0).

10.68 Sybase

[\[Notes en ligne\]](#)

10.68.1 sybase affected rows

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase affected rows](#)(int *link_identifieur*)

Retourne le nombre de lignes affectées par la dernière requête.

[sybase affected rows\(\)](#) retourne le nombre de lignes affectées par la dernière requête INSERT, UPDATE ou DELETE sur le serveur associé à l'identifiant de connexion *link_identifieur*. Si le lien n'est pas précisé, le dernier lien ouvert est utilisé.

Cette commande ne sert à rien sur les requête SELECT : uniquement sur des requêtes qui modifient les lignes. Pour connaître le nombre de lignes retournées par un SELECT, utilisez [sybase_num_rows\(\)](#). Note : *Cette fonction est disponible avec l'interface CT vers Sybase, mais pas avec la librairie DB.*

10.68.2 sybase close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [sybase close](#)(int *link_identifieur*)

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[sybase close\(\)](#) termine la connexion avec le serveur Sybase associé à l'identifiant de connexion *link_identifieur*.

Notez qu'il n'est pas utile de fermer les connexions non persistantes, car elles seront terminées à la fin du script.

[sybase close\(\)](#) ne ferme pas les connexions persistantes générées par [sybase_pconnect\(\)](#).

Voir aussi : [sybase_connect\(\)](#) et [sybase_pconnect\(\)](#).

10.68.3 sybase connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase connect](#)(string *servername*, string *username*, string *password*)

Retourne un identifiant positif de lien Sybase en cas de succès, et FALSE en cas d'erreur.

[sybase connect\(\)](#) établit une connexion à un serveur Sybase. Le nom de serveur *servername* doit être valide, défini dans le fichier d'interface.

Si un deuxième appel à [sybase connect\(\)](#) est fait avec les mêmes arguments, une nouvelle connexion ne sera pas établie, mais ce sera l'identifiant de la connexion déjà ouverte qui sera retourné.

La connexion sera fermée dès la fin du script, à moins qu'elle ne soit pas explicitement fermée avec [sybase_close\(\)](#).

Voir aussi [sybase_pconnect\(\)](#) et [sybase_close\(\)](#).

[10.68.4 sybase_data_seek](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [sybase_data_seek](#)(int *result_identif*ier, int *row_number*)

Retourne TRUE en cas de succès, et FALSE en cas d'échec.

[sybase_data_seek\(\)](#) déplace le pointeur interne de ligne du résultat Sybase associé à *result_identif*ier jusqu'à la ligne *row_number*. Le prochain appel à [sybase_fetch_row\(\)](#) sans préciser la ligne, retournera la ligne *row_number*.

Voir aussi: [sybase_data_seek\(\)](#).

[10.68.5 sybase_fetch_array](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [sybase_fetch_array](#)(int *result*)

Retourne un tableau qui contient la ligne demandée, ou FALSE s'il ne reste plus de ligne.

[sybase_fetch_array\(\)](#) est une version évoluée de [sybase_fetch_row\(\)](#). En plus d'enregistrer les données dans un tableau à index numérique, cette fonction peut aussi les enregistrer dans un tableau associatif, en utilisant les nom des champs comme clés.

Il est très important de noter que [sybase_fetch_array\(\)](#) N'est PAS nettement plus lent que [sybase_fetch_row\(\)](#), tandis qu'elle fourni un confort d'utilisation notable.

Pour plus de détails : [sybase_fetch_row\(\)](#).

[10.68.6 sybase_fetch_field](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [sybase_fetch_field](#)(int *result*, int *field_offset*)

Retourne un objet contenant les informations du champs.

[sybase_fetch_field\(\)](#) sert à obtenir des informations à propos des champs dans le résultat *result*. Si l'offset du champs n'est pas précisé, le champs suivant est traité.

Les propriétés des objets sont :

- name – column name. nom de la colonne. Si la colonne est un résultat de fonction, le nom de cette fonction devient computed#N, où #N est un numéro de série.
- column_source – la table d'origine de la colonne.
- max_length – taille maximale de la colonne
- numeric – 1 si la colonne est de type numérique.

Voir aussi [sybase_field_seek\(\)](#).

[10.68.7 sybase_fetch_object](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase_fetch_object](#)(int *result*)

Retourne un objet qui contient la ligne demandée, en cas de succès, et FALSE en cas d'erreur.

[sybase_fetch_object\(\)](#) est similaire à [sybase_fetch_array\(\)](#), avec une différence : c'est un objet qui est retourné à la place d'un tableau. Indirectement, cela signifie que vous ne pourrez accéder aux valeurs que par les propriétés, et non plus avec des offsets (les nombres sont interdits comme nom de propriété).

Au niveau de la vitesse, cette fonction est identique à [sybase_fetch_array\(\)](#), et presque aussi rapide que [sybase_fetch_row\(\)](#) (la différence est insignifiante).

Voir aussi: [sybase_fetch_array\(\)](#) et [sybase_fetch_row\(\)](#).

10.68.8 sybase_fetch_row

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [sybase_fetch_row](#)(int *result*)

Retourne un tableau qui contient la ligne demandée, en cas de succès, et FALSE en cas d'erreur.

[sybase_fetch_row\(\)](#) lit une ligne dans le résultat associé à l'identifiant de résultat *result*. La ligne retournée est sous la forme d'un tableau. Chaque champs est enregistré dans un index du tableau, les index commençant à 0.

Les prochains appels à [sybase_fetch_row\(\)](#) retourneront la ligne suivante du résultat, ou FALSE, si il ne reste plus de lignes.

Voir aussi: [sybase_fetch_array\(\)](#), [sybase_fetch_object\(\)](#), [sybase_data_seek\(\)](#) et [sybase_result\(\)](#).

10.68.9 sybase_field_seek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase_field_seek](#)(int *result*, int *field_offset*)

Modifie l'index d'un champs. Le prochain appel à la fonction [sybase_fetch_field\(\)](#) sans préciser l'index du champs retournera ce champs.

Voir aussi: [sybase_fetch_field\(\)](#).

10.68.10 sybase_free_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [sybase_free_result](#)(int *result*)

[sybase_free_result\(\)](#) n'est vraiment utile que si vous risquez d'utiliser trop de mémoire durant votre script. La mémoire occupée par les résultats est automatiquement libérée à la fin du script. Mais, si vous êtes sûr de ne pas avoir besoin du résultat ultérieurement.

10.68.11 sybase_get_last_message

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [sybase_get_last_message](#)(void)

[sybase_get_last_message\(\)](#) retourne le dernier message rapporté par le serveur.

[10.68.12 sybase_min_client_severity](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [sybase_min_client_severity](#)(int *severity*)

[sybase_min_client_severity\(\)](#) fixe la sévérité minimale du client. Note : *Cette fonction est disponible avec l'interface CT vers Sybase, mais pas avec la librairie DB.*

See also: [sybase_min_server_severity\(\)](#).

[10.68.13 sybase_min_error_severity](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [sybase_min_error_severity](#)(int *severity*)

[sybase_min_error_severity\(\)](#) fixe la sévérité minimale du client pour les erreurs.
Voir aussi: [sybase_min_message_severity\(\)](#).

[10.68.14 sybase_min_message_severity](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [sybase_min_message_severity](#)(int *severity*)

[sybase_min_message_severity\(\)](#) fixe la sévérité minimale du client pour les messages.
Voir aussi: [sybase_min_error_severity\(\)](#).

[10.68.15 sybase_min_server_severity](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [sybase_min_server_severity](#)(int *severity*)

[sybase_min_server_severity\(\)](#) fixe la sévérité minimale du client pour le serveur. Note : *Cette fonction est disponible avec l'interface CT vers Sybase, mais pas avec la librairie DB.*

Voir aussi: [sybase_min_client_severity\(\)](#).

[10.68.16 sybase_num_fields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase_num_fields](#)(int *result*)

[sybase_num_fields\(\)](#) retourne le nombre de champs dans un résultat.
Voir aussi: [sybase_query\(\)](#), [sybase_fetch_field\(\)](#) et [sybase_num_rows\(\)](#).

[10.68.17 sybase_num_rows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase_num_rows](#) (int *result*)

[sybase_num_rows\(\)](#) retourne le nombre de lignes dans un résultat.

Voir aussi: [sybase_query\(\)](#) et [sybase_fetch_row\(\)](#).

[10.68.18 sybase_pconnect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase_pconnect](#) (string *servername*, string *username*, string *password*)

Retourne un identifiant de connexion positive en cas de succès, et FALSE en cas d'erreur.

[sybase_pconnect\(\)](#) se comporte comme [sybase_connect\(\)](#) avec deux différences majeures :

Premièrement, lors de la connexion, la fonction va chercher une connexion (persistante) déjà ouverte, avec le même hôte, nom de compte et mot de passe. Si une telle connexion est trouvée, un identifiant de cette connexion est retourné, plutôt que d'en ouvrir une nouvelle.

Deuxièmement, la connexion au serveur SyBase ne sera pas terminée lors de la fin du script. Au contraire, le lien sera maintenu pour des connexions ultérieures. [sybase_close\(\)](#) ne fermera pas un lien créé par [sybase_pconnect\(\)](#).

Ce type de liens est dit 'persistant'.

[10.68.19 sybase_query](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase_query](#) (string *query*, int *link_identifier*)

Retourne un identifiant de résultat positif en cas de succès, et FALSE sinon.

[sybase_query\(\)](#) envoie une requête à la base de données courante, sur le serveur associé à l'identifiant de connexion. Si l'identifiant de connexion n'est pas précisé, la fonction essaiera d'utiliser la dernière connexion ouverte. Si aucune connexion n'a été ouverte, la fonction va tenter d'ouvrir une connexion avec la fonction [sybase_connect\(\)](#).

Voir aussi: [sybase_select_db\(\)](#) et [sybase_connect\(\)](#).

[10.68.20 sybase_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [sybase_result](#) (int *result*, int *i*, mixed *field*)

[sybase_result\(\)](#) retourne le contenu d'une cellule. L'argument *field* peut être l'index du champs, ou bien le nom du champs, ou encore, le nom de la table " point " le nom du champs. Si la colonne a été aliasée ('SELECT foo AS bar FROM...'), utilisez l'alias à la place du nom de la colonne.

Lorsque vous travaillez sur des résultats de grande taille, vous devriez utiliser les autres fonctions qui lisent une ligne entière (voir plus loin). Etant donné que ces fonctions lisent une ligne entière, elles sont BEAUCOUP plus rapide que [sybase_result\(\)](#). De plus, l'utilisation d'index numérique est beaucoup plus rapide que les noms des champs, ou les noms des tables et des champs.

Fonctions de substitution, à haute efficacité : [sybase_fetch_row\(\)](#), [sybase_fetch_array\(\)](#) et

[sybase_fetch_object\(\)](#).

10.68.21 sybase_select_db

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [sybase_select_db](#)(string *database_name*, int *link_identifier*)

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[sybase_select_db\(\)](#) change la base de données courante et active sur le serveur associé avec l'identifiant de connexion *link_identifier*. Si *link_identifier* n'est pas précisé, le dernier lien ouvert est utilisé. Si aucun lien n'a été ouvert, la fonction va tenter d'en établir un en appelant [sybase_connect\(\)](#).

Tous les prochains appels à [sybase_query\(\)](#) seront faites sur la base de données courante et active.

Voir aussi : [sybase_connect\(\)](#), [sybase_pconnect\(\)](#) et [sybase_query\(\)](#).

10.69 ODBC

[\[Notes en ligne\]](#)

10.69.1 odbc_autocommit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_autocommit](#)(int *connection_id*, int *OnOff*)

Sans paramètre *OnOff*, cette fonction retourne le statut d'auto-validation de la connexion *connection_id*. TRUE si le mode est activé, FALSE si il ne l'est pas, ou si une erreur survient.

Si *OnOff* vaut TRUE, l'auto-validation est activée. Si il est FALSE, l'auto-validation est désactivée.

Retourne TRUE en cas de succès, FALSE en cas d'échec.

Par défaut, l'auto-validation est activée. Désactiver l'auto-validation est équivalent à démarrer une transaction.

Voir aussi [odbc_commit\(\)](#) et [odbc_rollback\(\)](#).

10.69.2 odbc_binmode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_binmode](#)(int *result_id*, int *mode*)

Types ODBC SQL affectés: BINARY, VARBINARY, LONGVARBINARY.

- ODBC_BINMODE_PASSTHRU: Mode Passthru
- ODBC_BINMODE_RETURN: Retourne tel quel.
- ODBC_BINMODE_CONVERT: Converti en char et retourne la valeur.

Lorsqu'une donnée SQL est convertie en caractère C, les 8 bits du caractère source sont représentés par deux caractères ASCII. Ces caractères sont des représentations ASCII des nombres au format hexadécimal. Par exemple, le binaire 00000001 est converti en "01" et le binaire 11111111 est converti en "FF".

mode	longueur	résultat
------	----------	----------

ODBC_BINMODE_PASSTHRU	0	passthru
ODBC_BINMODE_RETURN	0	passthru
ODBC_BINMODE_CONVERT	0	passthru
ODBC_BINMODE_PASSTHRU	0	passthru
ODBC_BINMODE_PASSTHRU	>0	passthru
ODBC_BINMODE_RETURN	>0	Tel quel
ODBC_BINMODE_CONVERT	>0	Caractère

Si [odbc_fetch_into\(\)](#) est utilisé, passthru signifie qu'une chaîne vide sera retournée pour ces colonnes.

Si *result_id* vaut 0, ces paramètres seront appliqués aux nouveaux résultats. Note : *La valeur par défaut de 4096 est 4096 et les valeurs par défaut de odbc_binmode est ODBC_BINMODE_RETURN. La gestion des colonnes binaires est aussi modifié par [odbc_longreadlen\(\)](#).*

[10.69.3 odbc_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [odbc_close](#)(int *connection_id*)

[odbc_close\(\)](#) ferme la connexion avec une source de données, représentée par l'identifiant de connexion. Note : *Cette fonction échouera si il y a des transactions en cours sur cette connexion. Dans ce cas, la connexion restera ouverte.*

[10.69.4 odbc_close_all](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [odbc_close_all](#)

[odbc_close_all\(\)](#) ferme toutes les connexions ODBC à des sources de données. Note : *Cette fonction échouera si il y a des transactions en cours sur cette connexion. Dans ce cas, la connexion restera ouverte.*

[10.69.5 odbc_commit](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_commit](#)(int *connection_id*)

Retourne: TRUE en case de succès, FALSE en cas d'erreur. Toutes les connexions en cours sur *connection_id* sont validées. Toutes les connexions en cours sur *connection_id* sont validées.

[10.69.6 odbc_connect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_connect](#)(string *dsn*, string *user*, string *password*, int *cursor_type*)

Retourne un identifiant de connexion ODBC ou 0 (FALSE) en cas d'erreur.

L'identifiant de connexion retournée par cette fonction est nécessaire pour toutes les autres fonctions ODBC. Vous pouvez avoir de multiples connexions en même temps. Le quatrième paramètre fixe le type de pointeur de résultat utilisé pour cette connexion. Ce paramètre n'est généralement pas nécessaire, mais il peut être utile pour contourner certains problèmes ODBC.

Avec certains pilotes ODBC, l'exécution de procédures enregistrées complexes peut produire l'erreur suivante : "Cannot open a cursor on a stored procedure that has anything other than a single select statement in it", ce qui signifie : "Impossible de créer un pointeur de résultat dans une procédure enregistrée qui est réduite à une simple sélection (SELECT)). Utiliser l'option SQL_CUR_USE_ODBC permet d'éviter cette erreur. De plus, certains pilotes ne supportent le paramètre optionnel de numéro de ligne dans [odbc_fetch_row\(\)](#). SQL_CUR_USE_ODBC peut aussi permettre de résoudre ces problèmes.

Les constantes suivantes sont définies comme type de pointeur :

- SQL_CUR_USE_IF_NEEDED
- SQL_CUR_USE_ODBC
- SQL_CUR_USE_DRIVER
- SQL_CUR_DEFAULT

Pour les connexions persistantes, reportez vous à [odbc_pconnect\(\)](#).

[10.69.7 odbc_cursor](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [odbc_cursor](#)(int *result_id*)

[odbc_cursor\(\)](#) lit le pointeur de fiche courante (cursorname) pour le résultat *result_id*.

[10.69.8 odbc_do](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [odbc_do](#)(int *conn_id*, string *query*)

[odbc_do\(\)](#) exécute la requête *query* avec la connexion *conn_id*.

[10.69.9 odbc_exec](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_exec](#)(int *connection_id*, string *query_string*)

Retourne FALSE en cas d'erreur. Retourne un identifiant de résultat ODBC en cas d'exécution réussie.

[odbc_exec\(\)](#) envoie une commande SQL à la source de données représentée par *connection_id*. Ce paramètre doit être un identifiant valide de connexion, retourné par [odbc_connect\(\)](#) ou [odbc_pconnect\(\)](#).

Voir aussi : [odbc_prepare\(\)](#) et [odbc_execute\(\)](#) pour les exécutions multiples de requêtes SQL.

10.69.10 odbc_execute

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_execute](#)(int *result_id*, array *parameters_array*)

Exécute une requête SQL préparée par [odbc_prepare\(\)](#). Retourne TRUE en cas d'exécution réussie, et FALSE sinon. Le tableau de paramètres *parameters_array* ne sert que si vous avez besoin de paramètres votre requête.

10.69.11 odbc_fetch_into

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_fetch_into](#)(int *result_id*, int *rownumber*, array *result_array*)

Retourne le nombre de colonnes dans le résultat, ou FALSE en cas d'erreur. *result_array* doit avoir été passé par référence, mais il peut être de n'importe quel type, étant donné qu'il sera converti en tableau. Le tableau contiendra les valeurs des colonnes, ces dernières étant numérotées à partir de 0.

10.69.12 odbc_fetch_row

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_fetch_row](#)(int *result_id*, int *row_number*)

Si [odbc_fetch_row\(\)](#) a réussi, TRUE est retourné. Si il n'y avait plus de ligne, ou en cas d'erreur, FALSE est retourné.

[odbc_fetch_row\(\)](#) lit une ligne dans le résultat identifié par *result_id* et retourné par [odbc_do\(\)](#) ou [odbc_exec\(\)](#). Après [odbc_fetch_row\(\)](#), les champs seront accessibles avec la fonction [odbc_result\(\)](#).

Si *row_number* est omis, *row_number* va tenter de lire la prochaine ligne dans le résultat. Des appels répétés à [odbc_fetch_row\(\)](#) avec et sans paramètre *row_number* peuvent être combinés librement.

Pour passer en revue toutes les lignes d'un résultat plusieurs fois, vous pouvez appeler [odbc_fetch_row\(\)](#) avec *row_number* = 1, puis continue à appeler [odbc_fetch_row\(\)](#) sans le paramètre *row_number* pour passer en revue tout le résultat. Si un pilote ne supporte pas la lecture des lignes par numéro, le paramètre sera ignoré.

10.69.13 odbc_field_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [odbc_field_name](#)(int *result_id*, int *field_number*)

[odbc_field_name\(\)](#) lit le nom de la colonne dont l'index est *field_number*. La numérotation des champs commence à 1. FALSE est retourné en cas d'erreur.

10.69.14 odbc_field_type

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [odbc_field_type](#)(int *result_id*, int *field_number*)

[odbc_field_type\(\)](#) retourne le type de données SQL d'un champs, identifié par son index. La numérotation des

champs commence à 1.

[10.69.15 odbc_field_len](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_field_len](#)(int *result_id*, int *field_number*)

[odbc_field_len\(\)](#) retourne la longueur du champs référence par le nombre *field_number*, dans la connexion ODBC *result_id*. Les numéros de champs commencent à 1.

[10.69.16 odbc_free_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_free_result](#)(int *result_id*)

Retourne toujours TRUE.

[odbc_free_result\(\)](#) n'est nécessaire que si vous craignez d'utiliser trop de mémoire lors de l'exécution de votre script. Tous les résultats en mémoire seront libérés dès la fin du script. Mais, si vous êtes sûr que vous n'aurez plus besoin d'un résultat jusqu'à la fin de votre script, vous pouvez appeler [odbc_free_result\(\)](#), et la mémoire associée à *result_id* sera libérée.

Note : *Si auto-validation est désactivée (voir [odbc_autocommit\(\)](#)) et que vous appelez [odbc_free_result\(\)](#) avant de valider vos requêtes, toutes les transactions préparées seront annulées.*

[10.69.17 odbc_longreadlen](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_longreadlen](#)(int *result_id*, int *length*)

Types ODBC SQL affectés: LONG, LONGVARBINARY.

Le nombre d'octets retournés à PHP est contrôlé par le paramètre *length*. Si sa valeur est 0, les colonnes de type Long seront transformées en chaîne vide.

Note : *La gestion des types LONGVARBINARY est aussi affectée par [odbc_binmode\(\)](#).*

[10.69.18 odbc_num_fields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_num_fields](#)(int *result_id*)

[odbc_num_fields\(\)](#) retourne le nombre de colonnes dans un résultat ODBC. Cette fonction retournera -1 en cas d'erreur. L'argument est un identifiant de résultat valide, retourné par [odbc_exec\(\)](#).

[10.69.19 odbc_pconnect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_pconnect](#)(string *dsn*, string *user*, string *password*, int *cursor_type*)

Retourne un identifiant de connexion ODBC ou 0 (FALSE) en cas d'erreur. Cette fonction se comporte de manière similaire à [odbc_connect\(\)](#), mais la connexion ouverte n'est pas vraiment terminée lorsque le script est terminé. Les prochaines requêtes qui se feront sur une connexion dont les *dsn*, *user*, *password* sont les mêmes que celle-ci (avec [odbc_connect\(\)](#) et [odbc_pconnect\(\)](#)) réutiliseront la connexion ouverte.

Note : *Les connexions persistantes n'ont aucun effet si PHP est utilisé comme CGI.*

Pour plus de détails sur le paramètre optionnel *cursor_type*, voyez [odbc_connect\(\)](#). Pour plus de détails sur les connexions persistantes, reportez vous à la FAQ PHP.

10.69.20 odbc_prepare

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_prepare](#)(int *connection_id*, string *query_string*)

Prépare une commande pour l'exécution.

Retourne un identifiant de résultat ODBC si la commande SQL a été préparée avec succès. L'identifiant peut être utilisé plus tard pour exécuter la commande avec [odbc_execute\(\)](#).

10.69.21 odbc_num_rows

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_num_rows](#)(int *result_id*)

[odbc_num_rows\(\)](#) retourne le nombre de lignes dans un résultat ODBC. Cette fonction retournera -1 en cas d'erreur. Pour les commandes INSERT, UPDATE et DELETE, [odbc_num_rows\(\)](#) retourne le nombre de ligne affectées. Pour les commandes SELECT, ce PEUT le nombre de lignes disponibles, mais ce n'est pas certains.

Note: [odbc_num_rows\(\)](#) après un SELECT retournera -1 avec de nombreux pilotes.

10.69.22 odbc_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [odbc_result](#)(int *result_id*, mixed *field*)

Retourne le contenu d'un champs.

field peut être aussi bien un entier, contenant le numéro de colonne du champs, dans le résultat, ou bien une chaîne de caractère, qui représente le nom du champs. Par exemple:

```
$item_3 = odbc_result($Query_ID, 3 );
$item_val = odbc_result($Query_ID, "val");
```

Le premier appel à [odbc_result\(\)](#) retourne la valeur du troisième champs de la ligne courante, du résultat *result_id*. Le deuxième appel à [odbc_result\(\)](#) retourne la valeur du troisième champs dont le nom est "val" de la ligne courante, du résultat *result_id*. Une erreur survient si le paramètre de colonne est inférieur à 1, ou dépasse le nombre de colonnes du résultat. De la même manière, une erreur survient si le nom du champs passé ne correspond à aucun champs dans le résultat.

Les index de champs commencent à 1. Pour plus d'informations sur la façon de lire des colonnes de type binaire ou long, reportez vous à [odbc_binmode\(\)](#) et [odbc_longreadlen\(\)](#).

10.69.23 odbc_result_all

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_result_all](#) (int *result_id*, string *format*)

Retourne le nombre de lignes dans le résultat, ou FALSE en cas d'erreur.

[odbc_result_all\(\)](#) affiche toutes les lignes d'un résultat. L'affichage se fait au format HTML. Avec l'option *format*, il est possible de modifier l'aspect global de la table.

10.69.24 odbc_rollback

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_rollback](#) (int *connection_id*)

Annule toutes les transactions sur la connexion *connection_id*. Retourne TRUE en cas de succès, et FALSE en cas d'échec.

10.69.25 odbc_setoption

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_setoption](#) (int *id*, int *function*, int *option*, int *param*)

Cette fonction donne accès aux options ODBC pour une connexion particulière ou un résultat de requête. Elle a été écrite pour aider à la résolution de problème liés aux pilotes ODBC récalcitrants. Vous aurez sûrement à utiliser cette fonction si vous êtes un programmeur ODBC et que vous comprenez les divers effets des options disponibles. Vous aurez aussi besoin d'un bon manuel de référence pour comprendre les options et leur usage. Différentes versions de pilotes supportent différentes versions d'options.

Etant donné que les effets peuvent varier d'un pilote à l'autre, l'utilisation de cette fonction dans des scripts voués à être livrés au public est très fortement déconseillée. De plus, certaines options ODBC ne sont pas disponibles car elles doivent être fixées avant l'établissement de la connexion. Cependant, si dans un cas bien spécifique, cette fonction vous permet d'utiliser PHP sans que votre patron vous pousse à utiliser un produit commercial, alors cela n'a pas d'importance.

Id est un identifiant de connexion, ou un identifiant de résultat, pour lequel vous souhaitez modifier des options. Pour `SQLSetConnectOption()`, c'est un identifiant de connexion. Pour `SQLSetStmtOption()`, c'est un identifiant de résultat.

function est la fonction ODBC à utiliser. La valeur doit être de 1 pour utiliser `SQLSetConnectOption()` et 2 pour `SQLSetStmtOption()`.

Le paramètre *option* est l'option à modifier.

Le paramètre *param* est la valeur de l'option *option*. *ODBC Setoption Examples*

```
// 1. L'option 102 de SQLSetConnectOption() est SQL_AUTOCOMMIT.
// 1 de SQL_AUTOCOMMIT est SQL_AUTOCOMMIT_ON.
// Cet exemple a le meme effet que
// odbc_autocommit($conn, TRUE);
odbc_setoption ($conn, 1, 102, 1);
// 2. Option 0 de SQLSetStmtOption() est SQL_QUERY_TIMEOUT.
// Cet exemple fixe le délai d'expiration à 30 secondes.
$result = odbc_prepare ($conn, $sql);
```

```
odbc_setopt ($result, 2, 0, 30);
odbc_execute ($result);
```

[10.70 Fonctions URL](#)

[\[Notes en ligne\]](#)

[10.70.1 base64_decode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [base64_decode](#)(string *encoded_data*)

[base64_decode\(\)](#) décode *encoded_data* et retourne les données décodées. Les informations initiales peuvent être binaires.

Voir aussi : [base64_encode\(\)](#) et la RFC–2045 section 6.8.

[10.70.2 base64_encode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [base64_encode](#)(string *data*)

[base64_encode\(\)](#) retourne *data* encodé en base64. Cet encodage est fait pour permettre aux informations binaires d'être manipulées par les systèmes qui ne gèrent pas correctement les 8 bits, comme par exemple, les corps de mail.

Une chaîne encodée Base64 prend, grosso modo, 33% de plus que les données initiales.

Voir aussi: [base64_decode\(\)](#), [chunk_split\(\)](#), et la RFC–2045 section 6.8.

[10.70.3 parse_url](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [parse_url](#)(string *url*)

[parse_url\(\)](#) retourne un tableau associatif contenant les composants de l'URL. Les composants recherchés sont : "scheme", "host", "port", "user", "pass", "path", "query", et "fragment".

[10.70.4 urldecode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [urldecode](#)(string *str*)

Décode toutes les séquences %## et les remplace par leur valeur. La chaîne ainsi décodée est retournée.

Exemple avec [urldecode\(\)](#)

```
$a = split ('', $querystring);
```

```

$i = 0;
while ($i < count ($a)) {
    $b = split ('=', $a [$i]);
    echo 'Value for parameter ', htmlspecialchars (urldecode ($b [0])),
        ' is ', htmlspecialchars (urldecode ($b [1])), "<BR>";
    $i++;
}

```

Voir aussi [urlencode\(\)](#).

10.70.5 urlencode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [urlencode](#) (string *str*)

Retourne une chaîne dont les caractères non-alphanumériques (hormis `-._`) sont remplacés par des séquences commençant par un caractère pourcentage (%), suivi de deux chiffres hexadécimaux. Les espaces sont remplacés par des signes plus (+). Ce codage est celui qui est utilisé pour poster des informations dans les formulaires HTML. Le type MIME est `application/x-www-form-urlencoded`. Ce codage est différent de celui spécifié dans la RFC1738 (voir [rawurlencode\(\)](#)) : pour des raisons historiques, les espaces sont remplacés par des signes plus (+). Cette fonction est pratique pour transmettre des informations via une URL. C'est aussi un moyen de passer des informations d'une page à l'autre. *Exemple avec [urlencode\(\)](#)*

```

echo '<A HREF="mycgi?foo=' . urlencode ($userinput) . '>';

```

Voir aussi [urldecode\(\)](#).

10.71 Fonctions sur les variables

[\[Notes en ligne\]](#)

10.71.1 doubleval

[\[Notes en ligne\]](#) [\[Exemples\]](#)

double [doubleval](#) (mixed *var*)

Retourne la valeur numérique (double) de la variable *var*.
var peut être de type scalaire. Vous ne pouvez pas utiliser la fonction [doubleval\(\)](#) avec un tableau ou un objet.

```

$var = '122.34343The';
$double_value_of_var = doubleval($var);
print $double_value_of_var; // affiche 122.34343

```

Voir aussi [intval\(\)](#), [strval\(\)](#), [settype\(\)](#) et [9.8.6 Définition du type](#).

[10.71.2 empty](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [empty](#)(mixed *var*)

Retourne la valeur FALSE si la variable *var* est affectée ou bien a une valeur différente de 0; la valeur TRUE dans les autres cas.

```
$var = 0;
if (empty($var)) { #retourne TRUE
print 'soit $var vaut 0, soit il n'est pas défini';
}
if (!isset($var)) { // retourne FALSE
print '$var n'est pas définie';
}
```

Notez que cette fonction n'a pas de sens si elle est utilisée sur autre chose qu'une variable. i.e. empty (addslashes(\$name)) n'a pas de sens, car cela revient à vérifier une entité qui n'est pas une variable. Voir aussi [isset\(\)](#) et [unset\(\)](#).

[10.71.3 gettype](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gettype](#)(mixed *var*)

Retourne le type de la variable PHP *var*.

Les chaînes de caractères que peut retourner la fonction sont les suivantes :

- "integer"
- "double"
- "string"
- "array"
- "object"
- "unknown type"

Voir aussi [settype\(\)](#).

[10.71.4 intval](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [intval](#)(mixed *var*, int *base*)

Retourne la valeur numérique (integer) de la variable *var*, en convertissant la valeur dans la base spécifiée (par défaut en base 10).

var peut être de type scalaire. Vous ne pouvez pas utiliser la fonction [intval\(\)](#) avec un tableau ou un objet.

Voir aussi [doubleval\(\)](#), [strval\(\)](#), [settype\(\)](#) et [9.8.6 Définition du type](#).

[10.71.5 is_array](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [is_array](#)(mixed *var*)

Renvoie la valeur TRUE si la variable *var* est un tableau, FALSE sinon.

Voir aussi [is_double\(\)](#), [is_float\(\)](#), [is_int\(\)](#), [is_integer\(\)](#), [is_real\(\)](#), [is_string\(\)](#), [is_long\(\)](#), et [is_object\(\)](#).

[10.71.6 is_double](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [is_double](#)(mixed *var*)

Renvoie TRUE si la variable *var* est du type "double", FALSE sinon.

Voir aussi [is_array\(\)](#), [is_float\(\)](#), [is_int\(\)](#), [is_integer\(\)](#), [is_real\(\)](#), [is_string\(\)](#), [is_long\(\)](#), et [is_object\(\)](#).

[10.71.7 is_float](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [is_float](#)(mixed *var*)

Cette fonction est un alias de la fonction [is_double\(\)](#).

Voir aussi [is_double\(\)](#), [is_real\(\)](#), [is_int\(\)](#), [is_integer\(\)](#), [is_string\(\)](#), [is_object\(\)](#), [is_array\(\)](#), et [is_long\(\)](#).

[10.71.8 is_int](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [is_int](#)(mixed *var*)

Cette fonction est un alias de la fonction [is_long\(\)](#).

Voir aussi [is_double\(\)](#), [is_float\(\)](#), [is_integer\(\)](#), [is_string\(\)](#), [is_real\(\)](#), [is_object\(\)](#), [is_array\(\)](#), et [is_long\(\)](#).

[10.71.9 is_integer](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [is_integer](#)(mixed *var*)

Cette fonction est un alias de la fonction [is_long\(\)](#).

Voir aussi [is_double\(\)](#), [is_float\(\)](#), [is_int\(\)](#), [is_string\(\)](#), [is_real\(\)](#), [is_object\(\)](#), [is_array\(\)](#), et [is_long\(\)](#).

[10.71.10 is_long](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [is_long](#)(mixed *var*)

Renvoie TRUE si la variable *var* est du type integer (long), FALSE sinon.

Voir aussi [is_double\(\)](#), [is_float\(\)](#), [is_int\(\)](#), [is_real\(\)](#), [is_string\(\)](#), [is_object\(\)](#), [is_array\(\)](#), et [is_integer\(\)](#).

10.71.11 is_object

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [is_object](#) (mixed *var*)

Renvoie TRUE si la variable *var* est un objet, FALSE sinon.

Voir aussi [is_long\(\)](#), [is_int\(\)](#), [is_integer\(\)](#), [is_float\(\)](#), [is_double\(\)](#), [is_real\(\)](#), [is_string\(\)](#), et [is_array\(\)](#).

10.71.12 is_real

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [is_real](#) (mixed *var*)

Cette fonction est un alias de la fonction [is_double\(\)](#).

Voir aussi [is_long\(\)](#), [is_int\(\)](#), [is_integer\(\)](#), [is_float\(\)](#), [is_double\(\)](#), [is_object\(\)](#), [is_string\(\)](#), et [is_array\(\)](#).

10.71.13 is_string

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [is_string](#) (mixed *var*)

Renvoie TRUE si la variable *var* est du type "string", FALSE sinon.

Voir aussi [is_long\(\)](#), [is_int\(\)](#), [is_integer\(\)](#), [is_float\(\)](#), [is_double\(\)](#), [is_real\(\)](#), [is_object\(\)](#), et [is_array\(\)](#).

10.71.14 isset

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [isset](#) (mixed *var*)

Renvoie TRUE si la variable *var* est définie, FALSE sinon.

Si une variable a été désaffectée avec la fonction [unset\(\)](#), la fonction [isset\(\)](#) renverra FALSE.

```
$a = "test";
echo isset ($a); // TRUE
unset ($a);
echo isset ($a); // FALSE
```

Voir aussi [empty\(\)](#) et [unset\(\)](#).

10.71.15 print_r

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [print_r](#) (mixed *expression*)

Cette fonction affiche des informations à propos d'une variable, de manière à ce qu'elle soit lisible. Pour une

chaîne, un entier ou un double, la valeur sera elle même sera affichée. Pour les tableaux, les valeurs seront présentées dans un format qui montre les clés et les valeurs. Une notation similaire est disponible pour les objets.

Comparer [print_r\(\)](#) et [var_dump\(\)](#).

```
<?php
$a = array (1, 2, array ("a", "b", "c"));
print_r ($a);
?>
```

[10.71.16 settype](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [settype](#)(string *var*, string *type*)

Affecte le type *type* à la variable *var*.

Les valeurs possibles pour le paramètre *type* sont :

- "integer"
- "double"
- "string"
- "array"
- "object"

Renvoie TRUE en cas de succès, FALSE sinon.

Voir aussi [gettype\(\)](#).

[10.71.17 strval](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strval](#)(mixed *var*)

Retourne la valeur de la variable *var*, au format chaîne de caractères.

var peut être un scalaire. Vous ne pouvez pas utiliser la fonction [strval\(\)](#) avec des tableaux ou des objets.

Voir aussi [doubleval\(\)](#), [intval\(\)](#), [settype\(\)](#) et [9.8.6 Définition du type](#).

[10.71.18 unset](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [unset](#)(mixed *var*)

[unset\(\)](#) désaffecte la variable *var* et renvoie TRUE.

Exemple d'utilisation de la fonction [unset\(\)](#)

```
unset ($foo);
unset ($bar['quux']);
```

Voir aussi [isset\(\)](#) et [empty\(\)](#).

[10.71.19 var_dump](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [var_dump](#)(mixed *expression*)

Cette fonction retourne les informations structurées d'une variable, y compris son type et sa valeur. Les tableaux sont explorés récursivement, avec des indentations, pour mettre en valeur leur structure. Comparez [var_dump\(\)](#) et [print_r\(\)](#).

```
<pre>
<?php
    $a = array (1, 2, array ("a", "b", "c"));
var_dump ($a);
?>
</pre>
```

[10.72 Fonctions Vmailmgr](#)

[\[Notes en ligne\]](#)

Ces fonctions requièrent [qmail](#) et le [vmailmgr package](#) package de Bruce Guenter. Pour toutes les fonctions, les deux variables suivants sont définies de la façon suivante :
String vdomain : le nom de votre domaine virtuel (vdomain.com)
String basepwd : le mot de passe de l'utilisateur "réel" qui gère les utilisateurs virtuels.
Ces fonctions retournent un statut, qui correspond à ceux définis dans [response.h](#)

- 0 ok
- 1 mauvais
- 2 erreur
- 3 erreur de connexion

Problème connu : [vm_deluser\(\)](#) n'efface pas un utilisateur d'un dossier comme il devrait le faire. [vm_addalias\(\)](#) ne fonctionne actuellement pas correctement.

```
<?php
dl("php3_vmailmgr.so"); //charge la librairie
$vdomain="vdomain.com";
$basepwd="password";
?>
```

10.72.1 vm_adduser

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [vm_adduser](#) (string *vdomain*, string *basepwd*, string *newusername*, string *newuserpassword*)

Ajoute un nouvel utilisateur virtuel *newusername*, avec le mot de passe *newuserpassword*.

10.72.2 vm_addalias

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [vm_addalias](#) (string *vdomain*, string *basepwd*, string *username*, string *alias*)

Ajoute un nouvel alias vers un utilisateur virtuel. *username* est le nom du compte email, et *alias* est un alias pour cet utilisateur.

10.72.3 vm_passwd

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [vm_passwd](#) (string *vdomain*, string *username*, string *password*, string *newpassword*)

Change le mot de passe d'utilisateurs virtuels. *username* est le nom de compte email, *password* est l'ancien mot de passe de l'utilisateur, et *newpassword* son nouveau mot de passe.

10.72.4 vm_delalias

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [vm_delalias](#) (string *vdomain*, string *basepwd*, string *alias*)

Supprime un alias.

10.72.5 vm_deluser

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [vm_deluser](#) (string *vdomain*, string *username*)

Supprime un utilisateur virtuel.

10.73 WDDX functions

[\[Notes en ligne\]](#)

Ces fonctions opèrent avec l'aide de [WDDX](#).

Notez bien que toutes les fonctions qui enregistrent des données, utilisent le premier élément d'un tableau pour savoir si ce tableau doit être enregistré sous la forme d'un tableau, ou d'une structure. Si le premier

élément a une clé de type chaîne, le tableau sera enregistré sous la forme d'une structure, et sinon, sous la forme d'un tableau. *Enregistrer une valeur simple*

```
<?php
print wddx_serialize_value("Exemple de paquet de PHP à WDDX ", "Paquet PHP");
?>
```

Cet exemple va produire le résultat suivant :

```
<wddxPacket version='0.9'><header comment='Paquet PHP' /><data>
<string>Exemple de paquet de PHP à WDDX</string></data></wddxPacket>
```

Utilisation de paquets incrémentaux

```
<?php
$pi = 3.1415926;
$packet_id = wddx_packet_start("PHP");
wddx_add_vars($packet_id, "pi");
/* Supposons que $villes provient d'une base de données */
$cities = array("Paris", "Marseilles", "Lyon");
wddx_add_vars($packet_id, "villes");
$packet = wddx_packet_end($packet_id);
print $packet;
?>
```

Cet exemple donnera :

```
<wddxPacket version='0.9'><header comment='PHP' /><data><struct>
<var name='pi'><number>3.1415926</number></var><var name='cities'>
<array length='3'><string>Paris</string><string>Marseilles</string>
<string>Lyon</string></array></var></struct></data></wddxPacket>
```

10.73.1 wddx_serialize_value

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [wddx_serialize_value](#) (mixed *var*, string *comment*)

[wddx_serialize_value\(\)](#) sert à créer un paquet WDDX à partir d'une seule valeur. Cette fonction prend la valeur de *var*, et un argument optionnel *comment* qui apparaîtra dans l'entête du paquet, et retourne un paquet WDDX.

10.73.2 wddx_serialize_vars

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [wddx_serialize_vars](#) (mixed *var_name*, mixed ...)

[wddx_serialize_vars\(\)](#) sert à créer un paquet WDDX avec une structure qui contient la représentation des variables passées en arguments.

[wddx_serialize_vars\(\)](#) prend un nombre variable d'arguments, chacun d'entre eux pouvant être une chaîne contenant le nom d'une variable, ou un tableau de chaîne de nom de variable, ou même d'autres tableaux.

wddx_serialize_vars

```
<?php
$a = 1;
$b = 5.5;
$c = array("bleu", "orange", "violet");
$d = "colors";
$clvars = array("c", "d");
print wddx_serialize_vars("a", "b", $clvars);
?>
```

L'exemple ci-dessus donnera :

```
<wddxPacket version='0.9'><header/><data><struct><var name='a'><number>1</number></var>
<var name='b'><number>5.5</number></var><var name='c'><array length='3'>
<string>bleu</string><string>orange</string><string>violet</string></array></var>
<var name='d'><string>colors</string></var></struct></data></wddxPacket>
```

[10.73.3 wddx_packet_start](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [wddx_packet_start](#)(string *comment*)

[wddx_packet_start\(\)](#) sert à créer un nouveau paquet WDDX, pour pouvoir y faire des ajouts incrémentaux de variables. Cette fonction prend un argument optionel *comment* et retourne un identifiant de paquet, qui servira à d'autres fonctions. Elle va automatiquement créer une définition de structure dans le paquet, pour accueillir des variables.

[10.73.4 wddx_packet_end](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [wddx_packet_end](#)(int *packet_id*)

[wddx_packet_end\(\)](#)() clos un paquet WDDX repéré par son identifiant *packet_id*.

[10.73.5 wddx_add_vars](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

[wddx_add_vars](#)(int *packet_id*, mixed *name_var*, mixed ...)

[wddx_add_vars\(\)](#) sert à enregistrer les variables passées en argument, et les ajouter au paquet repéré par son identifiant *packet_id*. Les variables enregistrées sont spécifiées de la même façon que pour

[wddx_serialize_vars\(\)](#).

10.73.6 wddx_deserialize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [wddx_deserialize](#) (string *packet*)

[wddx_deserialize\(\)](#) prend une chaîne *packet* et la lit. Cette fonction retourne un résultat qui peut être une chaîne, un nombre ou un tableau. Notez que les structures sont lues sous la forme de tableau associatifs.

10.74 Analyseur syntaxique XML

[\[Notes en ligne\]](#)

10.74.1 Introduction

[\[Notes en ligne\]](#)

10.74.1.1 A propos de XML

[\[Notes en ligne\]](#)

Le langage XML (eXtensible Markup Language (Langage à Balises Etendu)) est un format structuré de données pour les échanges sur le web. C'est un standard défini par le consortium World Wide Web (W3C). Plus d'informations à propos du XML et des technologies afférentes sont accessibles (en anglais) <http://www.w3.org/XML/>.

10.74.1.2 Installation

[\[Notes en ligne\]](#)

Cette extension de PHP utilise *expat*, disponible à <http://www.jclark.com/xml/>. Le fichier Makefile livré avec expat ne construit pas par défaut de librairie : il faut utiliser la ligne suivante :

```
libexpat.a: $(OBJS)
ar -rc $ $(OBJS)
ranlib $
```

Les sources de expat sont disponibles à <http://www.guardian.no/~ssb/phpxml.html>.

Notez que si vous utilisez Apache-1.3.7 ou plus récent, vous disposez déjà de la librairie expat. Configurez simplement PHP avec `--with-xml` (sans aucun autre information) et la librairie expat d'Apache sera automatiquement utilisée.

Sous UNIX, lancez la configuration de PHP avec l'option `--with-xml`, la librairie expat étant installée là où votre compilateur peut la trouver. Si vous compilez PHP comme module de PHP 1.3.9 ou plus récent, PHP utilisera automatiquement le module *expat* livré avec Apache. Il vous faudra peut être fixer les valeurs des variables d'environnement *CPPFLAGS* et *LDFLAGS*, si vous avez fait une installation exotique.

Compilez PHP. *Tada!* C'est fait !

10.74.1.3 A propos de cette extension :

[\[Notes en ligne\]](#)

Cette extension PHP supporte la librairie *expat* de James Clark sous PHP. Cela vous permettra d'analyser mais pas de valider les documents XML. Il supporte trois types de codage différents, disponibles aussi sous

PHP: US-ASCII, ISO-8859-1 et UTF-8. UTF-16 n'est pas supporté.

Cette extension vous permet de créer des [10.74.3 xml_parser_create](#) puis de définir des *points d'entrée* pour chaque événement XML. Les analyseurs XML disposent de quelques [10.74.19 xml_parser_set_option](#).

Les gestionnaires d'évènements XML sont:

Fonction PHP de configuration du gestionnaire	Description de l'évènement
xml_set_element_handler()	Un événement est généré à chaque fois que l'analyseur XML rencontre une balise de début ou de fin. Deux gestionnaires sont disponibles : un pour le début, et un pour la fin. @tab
xml_set_character_data_handler() @tab "Character data" correspond grosso modo à tout ce qui n'est pas une balise XML, y compris les espaces entre les balises. Notez bien que l'analyseur XML n'ajoute ou n'efface aucun espace, et que c'est à l'application (c'est à dire vous) de décider de la signification de ces espaces. @tab	
xml_set_processing_instruction_handler() @tab Les programmeurs PHP sont habitués aux instructions exécutables (processing instructions ou PIs). <?php ?> est une instruction exécutable où php est appelé programme cible. Ces instructions sont gérées de manière spécifiques, (sauf le programme cible, qui est réservé à XML). @tab	
xml_set_default_handler()	Tout ce qui n'a pas trouvé de gestionnaire est transmis au gestionnaire par défaut. Vous retrouverez par exemple, les déclarations de type de document dans ce gestionnaire. @tab
xml_set_unparsed_entity_decl_handler() @tab Ce gestionnaire est appelé pour gérer les déclaration des entités non analysés. @tab	
xml_set_notation_decl_handler() @tab Ce gestionnaire est appelé pour gérer les notations. @tab	
xml_set_external_entity_ref_handler() @tab Ce gestionnaire est appelé lorsque l'analyseur XML trouve une référence à un fichier externe. Cela peut être un fichier, ou une URL. Reportez vous à 10.74.2.3 XML Entité externe pour un exemple. @tab	

[10.74.1.4 Problèmes de casse](#)

[\[Notes en ligne\]](#)

Les fonctions de gestion des balises peuvent rencontrer des balises en minuscule, majuscule ou encore dans un mélange des deux. En XML, la procédure standard est d' "identifier les séquences de caractère qui ne sont pas reconnues comme majuscule, et de les remplacer par leur équivalent majuscule". En d'autres termes, XML met toutes lettres en majuscules.

Par défaut, tous les noms des éléments qui sont transmis aux fonctions de gestion sont mises en majuscule. Ce comportement est contrôlé par l'analyseur XML, et peut être lu et modifié avec les fonctions respectives

[xml_parser_get_option\(\)](#) et [xml_parser_set_option\(\)](#), respectivement.

10.74.1.5 Error Codes

[\[Notes en ligne\]](#)

Les constantes suivantes sont définies comme des codes d'erreurs XML : (retournée par [xml_parse\(\)](#)):

- XML_ERROR_NONE
- XML_ERROR_NO_MEMORY
- XML_ERROR_SYNTAX
- XML_ERROR_NO_ELEMENTS
- XML_ERROR_INVALID_TOKEN
- XML_ERROR_UNCLOSED_TOKEN
- XML_ERROR_PARTIAL_CHAR
- XML_ERROR_TAG_MISMATCH
- XML_ERROR_DUPLICATE_ATTRIBUTE
- XML_ERROR_JUNK_AFTER_DOC_ELEMENT
- XML_ERROR_PARAM_ENTITY_REF
- XML_ERROR_UNDEFINED_ENTITY
- XML_ERROR_RECURSIVE_ENTITY_REF
- XML_ERROR_ASYNC_ENTITY
- XML_ERROR_BAD_CHAR_REF
- XML_ERROR_BINARY_ENTITY_REF
- XML_ERROR_ATTRIBUTE_EXTERNAL_ENTITY_REF
- XML_ERROR_MISPLACED_XML_PI
- XML_ERROR_UNKNOWN_ENCODING
- XML_ERROR_INCORRECT_ENCODING
- XML_ERROR_UNCLOSED_CDATA_SECTION
- XML_ERROR_EXTERNAL_ENTITY_HANDLING

10.74.1.6 Codage des caractères

[\[Notes en ligne\]](#)

L'extension XML de PHP supporte les caractères [Unicode](#) gr'ce à différents codages. Il y a deux types de codages de caractères : le codage à la source et le codage à la cible. PHP utilise le UTF-8 comme représentation interne.

L'encodage à la source est effectué lors de [10.74.12 xml_parse](#) du fichier par XML. Lors de la [10.74.3 xml_parser_create](#), un type de codage à la source doit être spécifié (et il ne pourra plus être modifié jusqu'à la destruction de l'analyseur). Les codages supportés sont : ISO-8859-1, US-ASCII et UTF-8. Les deux derniers sont des codages à un seul octet, c'est à dire que les caractères sont représentés sur un seul octet. UTF-8 peut représenter des caractères composés par un nombre variable de bits (jusqu'à 21), allant de 1 à quatre octets. Le codage par défaut utilisé par PHP ISO-8859-1.

Le codage à la cible est effectué lorsque PHP transfère les données aux gestionnaires XML. Lorsqu'un analyseur est créé, le codage à la cible est spécifié de la même façon que le codage à la source, mais il peut être modifié à tout moment. Le codage à la cible affectera les balises, tout comme les données brutes, et les noms des instructions exécutables.

Si l'analyseur XML rencontre un caractère qu'il ne connaît pas (hors limite, par exemple), il retournera une erreur.

Si PHP rencontre un caractère dans le document XML analysé, qu'il ne peut pas représenter dans le codage à

la cible choisi, le caractère sera remplacé par un point d'interrogation (cette attitude est susceptible de changer ultérieurement).

[10.74.2 Quelques exemples](#)

[\[Notes en ligne\]](#)

Voici une liste d'exemple de code PHP qui analyse un document XML.

[10.74.2.1 Exemple de structure XML](#)

[\[Notes en ligne\]](#)

Ce premier exemple affiche la structure de l'élément de début dans un document avec indentation/ *Afficher une structure XML*

```
$file = "data.xml";
$depth = array();
function startElement($parser, $name, $attrs) {
    global $depth;
    for ($i = 0; $i < $depth[$parser]; $i++) {
        print " ";
    }
    print "$name\n";
    $depth[$parser]++;
}
function endElement($parser, $name) {
    global $depth;
    $depth[$parser]--;
}
$xml_parser = xml_parser_create();
xml_set_element_handler($xml_parser, "startElement", "endElement");
if (!$fp = fopen($file, "r")) {
    die("could not open XML input");
}
while ($data = fread($fp, 4096)) {
    if (!xml_parse($xml_parser, $data, feof($fp))) {
        die(sprintf("XML error: %s at line %d",
            xml_error_string(xml_get_error_code($xml_parser)),
            xml_get_current_line_number($xml_parser)));
    }
}
xml_parser_free($xml_parser);
```

[10.74.2.2 XML Transtypage XML -> HTML](#)

[\[Notes en ligne\]](#)

XML Transtypage XML -> HTML

Cet exemple remplace les balises XML d'un document par des balises HTML. Les éléments inconnus seront ignorés. Bien entendu, cet exemple sera appliqué à un type précis de fichiers XML.

```

$file = "data.xml";
$map_array = array(
    "BOLD"      => "B",
    "EMPHASIS" => "I",
    "LITERAL"  => "TT"
);
function startElement($parser, $name, $attrs) {
global $map_array;
if ($htmltag = $map_array[$name]) {
print "<$htmltag>";
}
}
function endElement($parser, $name) {
global $map_array;
if ($htmltag = $map_array[$name]) {
print "</$htmltag>";
}
}
function characterData($parser, $data) {
print $data;
}
$xml_parser = xml_parser_create();
// use case-folding so we are sure to find the tag in $map_array
xml_parser_set_option($xml_parser, XML_OPTION_CASE_FOLDING, TRUE);
xml_set_element_handler($xml_parser, "startElement", "endElement");
xml_set_character_data_handler($xml_parser, "characterData");
if (!$fp = fopen($file, "r")) {
die("could not open XML input");
}
while ($data = fread($fp, 4096)) {
if (!xml_parse($xml_parser, $data, feof($fp))) {
die(sprintf("XML error: %s at line %d",
xml_error_string(xml_get_error_code($xml_parser)),
xml_get_current_line_number($xml_parser)));
}
}
xml_parser_free($xml_parser);

```

[10.74.2.3 XML Entité externe](#)

[\[Notes en ligne\]](#)

Cet exemple exploite les références externes de XML : il est possible d'utiliser un gestionnaire d'entité externe pour inclure et analyser les documents, tous comme les instructions exécutables peuvent servir à inclure et analyser d'autres documents, et aussi fournir une indication de confiance (voir plus bas). Le document XML qui est utilisé dans cet exemple est fourni plus loin dans l'exemple ('xmltest.xml' et 'xmltest2.xml').

Entité externe

```

$file = "xmltest.xml";
function trustedFile($file) {
    // only trust local files owned by ourselves
    if (!eregi("^[a-z]+://", $file)
        && fileowner($file) == getmyuid()) {

```

```

return TRUE;
    }
return FALSE;
}
function startElement($parser, $name, $attribs) {
print "&lt;<font color=\"#0000cc\">$name</font>";
if (sizeof($attribs)) {
while (list($k, $v) = each($attribs)) {
print " <font color=\"#009900\">$k</font>=\"<font
color=\"#990000\">$v</font>\"";
    }
}
print "&gt;";
}
function endElement($parser, $name) {
print "&lt;/<font color=\"#0000cc\">$name</font>&gt;";
}
function characterData($parser, $data) {
print "<b>$data</b>";
}
function PIHandler($parser, $target, $data) {
switch (strtolower($target)) {
case "php":
global $parser_file;
    // If the parsed document is "trusted", we say it is safe
    // to execute PHP code inside it. If not, display the code
    // instead.
if (trustedFile($parser_file[$parser])) {
eval($data);
    } else {
printf("Untrusted PHP code: <i>%s</i>",
htmlspecialchars($data));
    }
break;
    }
}
function defaultHandler($parser, $data) {
if (substr($data, 0, 1) == "&" && substr($data, -1, 1) == ";") {
printf('<font color="#aa00aa">%s</font>',
htmlspecialchars($data));
    } else {
printf('<font size="-1">%s</font>',
htmlspecialchars($data));
    }
}
function externalEntityRefHandler($parser, $openEntityNames, $base, $systemId,
    $publicId) {
if ($systemId) {
if (!list($parser, $fp) = new_xml_parser($systemId)) {
printf("Could not open entity %s at %s\n", $openEntityNames,
    $systemId);
return FALSE;
    }
while ($data = fread($fp, 4096)) {
if (!xml_parse($parser, $data, feof($fp))) {
printf("XML error: %s at line %d while parsing entity %s\n",
xml_error_string(xml_get_error_code($parser)),
xml_get_current_line_number($parser), $openEntityNames);
xml_parser_free($parser);
return FALSE;
    }
}
}
}
}

```

```

xml_parser_free($parser);
return TRUE;
    }
return FALSE;
}
function new_xml_parser($file) {
global $parser_file;
    $xml_parser = xml_parser_create();
xml_parser_set_option($xml_parser, XML_OPTION_CASE_FOLDING, 1);
xml_set_element_handler($xml_parser, "startElement", "endElement");
xml_set_character_data_handler($xml_parser, "characterData");
xml_set_processing_instruction_handler($xml_parser, "PIHandler");
xml_set_default_handler($xml_parser, "defaultHandler");
xml_set_external_entity_ref_handler($xml_parser, "externalEntityRefHandler");
if (!$fp = @fopen($file, "r")) {
return FALSE;
    }
if (!is_array($parser_file)) {
settype($parser_file, "array");
    }
    $parser_file[$xml_parser] = $file;
return array($xml_parser, $fp);
}
if (!(list($xml_parser, $fp) = new_xml_parser($file))) {
die("could not open XML input");
}
print "<pre>";
while ($data = fread($fp, 4096)) {
if (!$xml_parse($xml_parser, $data, feof($fp))) {
die(sprintf("XML error: %s at line %d\n",
xml_error_string(xml_get_error_code($xml_parser)),
xml_get_current_line_number($xml_parser)));
    }
}
print "</pre>";
print "parse complete\n";
xml_parser_free($xml_parser);
?>

```

xmltest.xml

```

<?xml version='1.0'?>
<!DOCTYPE chapter SYSTEM "/just/a/test.dtd" [
<!ENTITY plainEntity "FOO entity">
<!ENTITY systemEntity SYSTEM "xmltest2.xml">
]>
<chapter>
  <TITLE>Title &plainEntity;</TITLE>
  <para>
    <informaltable>
      <tgroup cols="3">
        <tbody>
          <row><entry>a1</entry><entry morerows="1">b1</entry><entry>c1</entry></row>
          <row><entry>a2</entry><entry>c2</entry></row>
          <row><entry>a3</entry><entry>b3</entry><entry>c3</entry></row>
        </tbody>
      </tgroup>
    </informaltable>

```

```

</para>
&systemEntity;
<sect1 id="about">
  <title>About this Document</title>
  <para>
    <!-- this is a comment -->
    <?php print 'Hi! This is PHP version ' .phpversion(); ?>
  </para>
</sect1>
</chapter>

```

This file is included from `xmltest.xml` : *xmltest2.xml*

```

<?xml version="1.0"?>
<!DOCTYPE foo [
<!ENTITY testEnt "test entity">
]>
<foo>
  <element attrib="value"/>
  &testEnt;
  <?php print "This is some more PHP code being executed."; ?>
</foo>

```

[10.74.3 xml_parser_create](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_parser_create](#) (string *encoding*)

encoding (optional)

- Le codage de caractère de l'analyseur : les codages suivants sont supportés :
 - ◆ ISO-8859-1 (par défaut)
 - ◆ US-ASCII
 - ◆ UTF-8

Cette fonction crée un analyseur XML et retourne une référence sur cet analyseur pour qu'il puisse être utilisé ultérieurement par d'autres fonctions XML. Retourne TRUE ou FALSE.

[10.74.4 xml_set_object](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [xml_set_object](#) (int *parser*, object *&object*)

Cette fonction rend l'analyseur *parser* utilisable depuis un objet. Toutes les méthodes de callback, affectées par [xml_set_element_handler\(\)](#), seront les méthodes de cet objet.

```
<?php
```

```

class xml {
var $parser;
function xml() {
    $this->parser = xml_parser_create();
xml_set_object($this->parser, this);
xml_set_element_handler($this->parser, "tag_open", "tag_close");
xml_set_character_data_handler($this->parser, "cdata");
}
function parse($data) {
xml_parse($this->parser, $data);
}
function tag_open($parser, $tag, $attributes) {
var_dump($parser, $tag, $attributes);
}
function cdata($parser, $cdata) {
var_dump($parser, $cdata);
}
function tag_close($parser, $tag) {
var_dump($parser, $tag);
}
} // fin de la classe xml
$xml_parser = new xml();
$xml_parser->parse(" <A ID=\"hallo\">PHP</A>");
?>

```

Note : [xml_set_object\(\)](#) a été ajouté dans PHP 4.0.

[10.74.5 xml_set_element_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_set_element_handler](#) (int *parser*, string *startElementHandler* , string *endElementHandler*)

Affecte les gestionnaires de début et de fin de l'analyseur XML *parser*. *startElementHandler* et *endElementHandler* sont des chaînes qui contiennent les noms de fonctions qui existent lorsque [xml_parse\(\)](#) est appelé pour créer *parser*.

La fonction *startElementHandler* doit accepter trois paramètres: *startElementHandler* (int *parser*, string *name*, string *attrs*)

parser

- Le premier paramètre, *parser*, est une référence sur l'analyseur XML qui appelle cette fonction.

name

- Le deuxième paramètre, *name*, contient le nom de l'élément qui a provoqué l'appel du gestionnaire. Si l'analyseur gère la [10.74.1.4 Problèmes de casse](#), cet élément sera en majuscule.

attrs

- Le troisième paramètre, *attrs*, contient un tableau associatif avec les attributs de l'éléments (si il en existe). Les clés de ce tableau seront les noms des attributs, et les valeurs seront les valeurs correspondantes des attributs. Les noms des attributs seront mis en majuscule si l'analyseur gère la [10.74.1.4 Problèmes de casse](#). Les valeurs des attributs seront intouchées. L'ordre original des attributs peut être retrouvé en passant en revue le tableau *attrs*, avec la fonction [each\(\)](#). La première clé sera la première clé du tableau.

La fonction *endElementHandler* doit accepter deux paramètres: *endElementHandler* (int *parser*, string *name*)

parser

- Le premier paramètre, *parser*, est une référence sur l'analyseur XML qui appelle cette fonction.

name

- Le second paramètre, *name*, contient le nom de l'élément qui a provoqué l'appel du gestionnaire. Si l'analyseur gère la [10.74.1.4 Problèmes de casse](#), cet élément sera en majuscule.

Si un gestionnaire reçoit une chaîne vide, ou FALSE, c'est qu'il est en train d'être désactivé. Retourne TRUE si le gestionnaire est actif, et FALSE sinon, ou si *parser* n'est pas un analyseur. Il n'est pas pour l'instant possible d'utiliser des objets pour servir de gestionnaire.

[10.74.6 xml_set_character_data_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_set_character_data_handler](#) (int *parser*, string *handler*)

Affecte les gestionnaires de début et de fin de l'analyseur XML *parser*. *handler* est une chaîne qui contient le nom d'une fonction qui existe lorsque [xml_parse\(\)](#) est appelé pour créer *parser*.

La fonction *handler* doit accepter deux paramètres: *handler* (int *parser*, string *data*)

parser

- Le premier paramètre, *parser*, est une référence sur l'analyseur XML qui appelle cette fonction.

data

- Le second paramètre, *data*, contient les caractères sous la forme d'une chaîne.

Si un gestionnaire reçoit une chaîne vide ou FALSE, c'est qu'il est en train d'être désactivé. Retourne TRUE si le gestionnaire est actif, et FALSE sinon, ou si *parser* n'est pas un analyseur. Il n'est pas pour l'instant possible d'utiliser des objets pour servir de gestionnaire.

[10.74.7 xml_set_processing_instruction_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_set_processing_instruction_handler](#) (int *parser*, string *handler*)

Affecte les gestionnaires d'instructions exécutables de l'analyseur XML *parser*. *handler* est une chaîne qui contient le nom d'une fonction qui existe lorsque [xml_parse\(\)](#) est appelé pour créer *parser*.

Une instruction exécutable a la forme suivante ::

```
<?
target
data?>
```

Vous pouvez mettre du code PHP entre ces balises, mais soyez conscient d'une des limitations des instructions exécutables de XML : la balise de fin d'instruction exécutable (?>) ne peut être échappée, ce qui fait que cette séquence NE DOIT JAMAIS apparaître dans le code PHP placé dans le document PHP. Si un tel texte apparaît, la balise de fin d'instruction exécutable sera reconnue, et le reste du code sera considéré comme des données brutes (et donc, pas exécutées).

La fonction *handler* doit accepter trois paramètres: *handler* (int *parser*, string *target*, string *data*)

parser

- Le premier paramètre, *parser*, est une référence sur l'analyseur XML qui appelle cette fonction.

target

- Le second paramètre, *target*, contient l'application cible.

data

- Le troisième paramètre, *data*, contient le code sous la forme d'une chaîne.

Si un gestionnaire reçoit une chaîne vide, ou FALSE, c'est qu'il est en train d'être désactivé. Retourne TRUE si le gestionnaire est actif, et FALSE sinon, ou si *parser* n'est pas un analyseur. Il n'est pas pour l'instant possible d'utiliser des objets pour servir de gestionnaire.

[10.74.8 xml_set_default_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_set_default_handler](#) (int *parser*, string *handler*)

Affecte le gestionnaire par défaut de l'analyseur XML *parser*. *handler* est une chaîne qui contient le nom d'une fonction qui existe lorsque [xml_parse\(\)](#) est appelé pour créer *parser*.

La fonction *handler* doit accepter deux paramètres: *handler* (int *parser*, string *data*)

parser

- Le premier paramètre, *parser*, est une référence sur l'analyseur XML qui appelle cette fonction.

data

- Le second paramètre, *data*, contient les caractères sous la forme d'une chaîne. Cela peut être une déclaration XML, un type de document, une entité ou d'autre données pour qui aucun gestionnaire n'est prévu.

Si un gestionnaire reçoit une chaîne vide ou FALSE, c'est qu'il est en train d'être désactivé. Retourne TRUE si le gestionnaire est actif, et FALSE sinon, ou si *parser* n'est pas un analyseur. Il n'est pas pour l'instant possible d'utiliser des objets pour servir de gestionnaire.

[10.74.9 xml_set_unparsed_entity_decl_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_set_unparsed_entity_decl_handler](#) (int *parser*, string *handler*)

Affecte les gestionnaires d'entité non déclaré de l'analyseur XML *parser*. *handler* est une chaîne qui contient le nom d'une fonction qui existe lorsque [xml_parse\(\)](#) est appelé pour créer *parser*.

Ce gestionnaire sera appelé si l'analyseur XML rencontre une déclaration d'entité externe avec une déclaration de NDATA, comme suit :

```
<!ENTITY name {publicId | systemId}
NDATA notationName>
```

Reportez vous à la [section 4.2.2 des spécifications XML 1.0](#) pour connaître les notations des entités externes. La fonction *handler* doit accepter six paramètres: *handler* (int *parser*, string *entityName*, string *base*, string *systemId*, string *publicId*, string *notationName*)

parser

- Le premier paramètre, *parser*, est une référence sur l'analyseur XML qui appelle cette fonction.
- *entityName*
Le nom de l'entité qui va être définie
- *base*
La meilleure base de résolution de l'identifiant système de cette entité externe. Actuellement, ce paramètre est toujours une chaîne vide.
- *systemId*
Identifiant système pour cet entité externe.
- *publicId*
Identifiant public pour cet entité externe.
- *notationName*
Nom de la notation de cette entité. (Voir [xml_set_notation_decl_handler\(\)](#)).

Si un gestionnaire reçoit une chaîne vide ou FALSE, c'est qu'il est en train d'être désactivé. Retourne TRUE si le gestionnaire est actif, et FALSE sinon, ou si *parser* n'est pas un analyseur. Il n'est pas pour l'instant possible d'utiliser des objets pour servir de gestionnaire.

[10.74.10 xml_set_notation_decl_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_set_notation_decl_handler](#) (int *parser*, string *handler*)

Affecte les gestionnaires de début et de fin de l'analyseur XML *parser*. *handler* est une chaîne qui contient le nom d'une fonction qui existe lorsque [xml_parse\(\)](#) est appelé pour créer *parser*. Une notation est une partie du DTD du document, qui a le format suivant :

```
<!NOTATION
name {systemId |
publicId}>
```

Reportez vous à la [section 4.2.2 des spécifications XML 1.0](#) pour connaître les notations des entités externes. La fonction *handler* doit accepter cinq paramètres: *handler* (int *parser*, string *notationName*, string *base*, string *systemId*, string *publicId*)

parser

- Le premier paramètre, *parser*, est une référence sur l'analyseur XML qui appelle cette fonction. ***notationName***
- Le nom de la notation, *name*, comme précisé dans le format de notation ci dessus. ***base***
- La meilleure base de résolution de l'identifiant système de cette entité externe. Actuellement, ce paramètre est toujours une chaîne vide. ***systemId***
- Identifiant système pour cette entité externe. ***publicId***
- Identifiant public pour cet entité externe.

Si un gestionnaire reçoit une chaîne vide ou FALSE, c'est qu'il est en train d'être désactivé. Retourne TRUE si le gestionnaire est actif, et FALSE sinon ou si *parser* n'est pas un analyseur. Il n'est pas pour l'instant possible d'utiliser des objets pour servir de gestionnaires.

10.74.11 xml_set_external_entity_ref_handler

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_set_external_entity_ref_handler](#) (int *parser*, string *handler*)

Fixe le gestionnaire d'entité externe de l'analyseur XML *parser*. *handler* et *endElementHandler* sont des chaînes qui contiennent les noms de fonction qui existent lorsque [xml_parse\(\)](#) est appelé pour créer le *parser*. La fonction *handler* doit accepter 5 paramètres, et retourner un entier. Si la valeur retourné par le gestionnaire est FALSE (comme par exemple si aucune valeur n'est retournée), l'analyseur XML s'arrêtera, et la fonction [xml_get_error_code\(\)](#) retournera XML_ERROR_EXTERNAL_ENTITY_HANDLING. **int** *handler* (int *parser*, string *openEntityNames* , string *base*, string *systemId*, string *publicId*)

parser

- Le premier paramètre, *parser*, est une référence sur l'analyseur XML qui appelle cette fonction. ***openEntityNames***
- Le deuxième paramètre, *openEntityNames*, est i.e. liste de noms d'entité, séparés par des espaces. Ces entités sont accessibles à l'analyse par cet entité (y compris le nom de l'entité référencé). ***base***
- La meilleure base de résolution de l'identifiant système de cet entité externe. Actuellement, ce paramètre est toujours une chaîne vide. ***systemId***
- Identifiant système pour cet entité externe. ***publicId***
- Le cinquième paramètre, *publicId*, est l'identifiant public, comme spécifié dans la déclaration d'entité, ou un chaîne vide, si aucune déclaration n'a été spécifiée. L'espace dans l'identifiant public sera normalisé comme spécifié dans les spécifications XML.

Si un gestionnaire reçoit une chaîne vide, ou FALSE, c'est qu'il est en train d'être désactivé. Retourne TRUE si le gestionnaire est actif, et FALSE sinon ou si *parser* n'est pas un analyseur. Il n'est pas pour l'instant possible d'utiliser des objets pour servir de gestionnaire.

10.74.12 xml_parse

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_parse](#)(int *parser*, string *data*, int *isFinal*)

parser

- une référence sur l'analyseur XML à utiliser.

data

- Une partie des données à analyser. Un document peut être analysé morceau par morceau, en appelant [xml_parse\(\)](#) plusieurs fois, tant que le paramètre *isFinal* est mis à TRUE pour le dernier morceau.

isFinal (optional)

- Si il vaut TRUE, *data* est la dernière partie à analyser.

Lorsqu'un document XML est analysé, les gestionnaires d'événements sont appelés aussi souvent que nécessaire, et retourne TRUE ou FALSE.

TRUE est retourné lorsque l'analyse a été concluante, et FALSE en cas d'échec, ou si *parser* n'est pas un analyseur valide. Lors d'un échec d'analyse, la cause de l'erreur peut être obtenue grâce aux fonctions [xml_get_error_code\(\)](#), [xml_error_string\(\)](#), [xml_get_current_line_number\(\)](#), [xml_get_current_column_number\(\)](#) et [xml_get_current_byte_index\(\)](#).

10.74.13 xml_get_error_code

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_get_error_code](#)(int *parser*)

parser

- Cette fonction retourne FALSE si *parser* n'est pas valide, ou sinon, retourne le numéro de colonne courante de la ligne courante de l'analyseur, qui correspond à la position d'analyse courante de l'analyseur XML.

Cette fonction retourne FALSE si *parser* n'est pas valide, ou sinon, retourne le numéro de colonne courante de la ligne courante de l'analyseur, qui correspond à la position d'analyse courante de l'analyseur XML.

10.74.14 xml_error_string

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [xml_error_string](#)(int *code*)

code

- Un message d'erreur, issu de [xml_get_error_code\(\)](#).

Retourne la chaîne avec un message textuel, décrivant l'erreur *code*, ou FALSE si aucune description n'a été

trouvée.

[10.74.15 xml_get_current_line_number](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_get_current_line_number](#)(int *parser*)

parser

- Une référence sur un analyseur XML valide.

Cette fonction retourne FALSE si *parser* n'est pas valide, ou sinon, retourne le numéro de la ligne en cours d'analyse.

[10.74.16 xml_get_current_column_number](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_get_current_column_number](#)(int *parser*)

parser

- Une référence sur un analyseur XML valide.

Cette fonction retourne FALSE si *parser* n'est pas valide, ou sinon, retourne le numéro de colonne courante de la ligne courante de l'analyseur, qui correspond à la position d'analyse courante de l'analyseur XML.

[10.74.17 xml_get_current_byte_index](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_get_current_byte_index](#)(int *parser*)

parser

- Une référence sur un analyseur XML valide.

Cette fonction retourne FALSE si *parser* n'est pas valide, ou sinon, retourne l'index de l'octet d'analyse courante de l'analyseur XML.

[10.74.18 xml_parser_free](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [xml_parser_free](#)(int *parser*)

parser

- Une référence sur un analyseur XML.

Cette fonction retourne FALSE si *parser* n'est pas une référence valide, ou sinon, détruit l'analyseur et retourne TRUE.

[10.74.19 xml_parser_set_option](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_parser_set_option](#) (int *parser*, int *option*, mixed *value*)

parser

- Une référence vers un analyseur XML.

option

- L'option à modifier. Voir ci dessous :

value

- La nouvelle valeur de l'option.

Cette fonction retourne FALSE si *parser* n'est pas une référence valide sur un analyseur XML, ou si l'option n'a pas pu être modifiée. Sinon, l'option est effectivement modifiée, et la fonction retourne TRUE.

Les options suivantes sont disponibles :

Option	Type de données	Description
XML_OPTION_CASE_FOLDING	entier	Contrôle la gestion de la 10.74.1.4 Problèmes de casse des balises de cet analyseur XML. Par défaut, activé. @tab
XML_OPTION_TARGET_ENCODING	string	Modifie le 10.74.1.6 Codage des caractères utilisé par cet analyseur XML. Par défaut, c'est celui qui a été spécifié lors de l'appel de xml_parser_create() . Les codages supportés sont ISO-8859-1, US-ASCII et UTF-8. @tab

[10.74.20 xml_parser_get_option](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [xml_parser_get_option](#) (int *parser*, int *option*)

parser

- Une référence sur un analyseur XML valide.
option
- L'option demandée. Reportez vous à [xml_parser_set_option\(\)](#) pour avoir la liste des options disponibles

Cette fonction retourne FALSE si *parser* n'est pas valide, ou sinon, retourne la valeur de l'option demandée. Reportez vous à [xml_parser_set_option\(\)](#) pour avoir la liste des options disponibles

10.74.21 utf8_decode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [utf8_decode](#)(string *data*)

Cette fonction décode la chaîne *data*, en supposant qu'elle est au format UTF-8, et la convertit au format ISO-8859-1.

Voir aussi [utf8_encode\(\)](#) pour plus de détails sur le codage UTF-8.

10.74.22 utf8_encode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [utf8_encode](#)(string *data*)

Cette fonction code la chaîne *data* au format UTF-8, et retourne la version codée. UTF-8 est un mécanisme standardisé utilisé par Unicode pour coder les caractères de grande taille dans des flots d'octets. UTF-8 est transparent pour les caractères ASCII, il est auto-synchronisé (c'est à dire qu'un programme peut toujours savoir dans un flot d'octet où un caractère commence), et peut être utilisé pour faire des comparaisons de chaînes standard, comme pour le tri. PHP utilise l'UTF-8 pour coder les caractères jusqu'à 4 octets comme ceci :

octets	bits	représentation
1	7	0bbbbbbb
2	11	110bbbb 10bbbbbb
3	16	1110bbbb 10bbbbbb 10bbbbbb
4	21	11110bbb 10bbbbbb 10bbbbbb 10bbbbbb

Chaque *b* représente un bit qui peut être utilisé pour enregistrer un caractère.

10.75 YAZ

[\[Notes en ligne\]](#)

The `@xref{function.yaz , , yaz() }` functions wrap the YAZ API. The home page of the project is <http://www.indexdata.dk/yaz/>. Information about the phpyaz module can be found at <http://www.indexdata.dk/phpyaz/>.

PHP/YAZ is much simpler to use than the C API for YAZ but less flexible. The intent is to make it easy to build basic client functions. It supports persistent stateless connections very similar to those offered by the

various SQL APIs that are available for PHP. This means that sessions are stateless but shared amongst users, thus saving the connect and INIT steps in many cases.

Before compiling PHP with the PHP/YAZ module you'll need the YAZ toolkit. Build YAZ and install it. Build PHP with your favourite modules and add option `--with-yaz`. Your task is roughly the following:

```
gunzip -c yaz-1.6.tar.gz|tar xf -
gunzip -c php-4.0.X.tar.gz|tar xf -
cd yaz-1.6
./configure --prefix=/usr
make
make install
cd ../php-4.0.X
./configure --with-yaz=/usr/bin
make
make install
```

PHP/YAZ keeps track of connections with targets (Z-Associations). A positive integer represents the ID of a particular association.

The script below demonstrates the parallel searching feature of the API. When invoked it either prints a query form (if no arguments are supplied) or if there are arguments (term and one or more hosts) it searches the targets in array host.

@xreffunction.yaz , , yaz()

```
$num_hosts = count ($host);
if (empty($term) || count($host) == 0) {
echo '<form method="get">
  <input type="checkbox"
name="host[]" value="bagel.indexdata.dk/gils">
GILS test
  <input type="checkbox"
name="host[]" value="localhost:9999/Default">
local test
  <input type="checkbox" checked="1"
name="host[]" value="z3950.bell-labs.com/books">
BELL Labs Library
<br>
RPN Query:
  <input type="text" size="30" name="term">
  <input type="submit" name="action" value="Search">
  ';
} else {
echo 'You searched for ' . htmlspecialchars($term) . '<br>';
for ($i = 0; $i > $num_hosts; $i++) {
  $id[] = yaz_connect($host[$i]);
yaz_syntax($id[$i], "sutrs");
yaz_search($id[$i], "rpn", $term);
}
yaz_wait();
for ($i = 0; $i < $num_hosts; $i++) {
echo '<hr>' . $host[$i] . " : ";
  $error = yaz_error($id[$i]);
if (!empty($error)) {
echo "Error: $error";
} else {
  $hits = yaz_hits($id[$i]);
echo "Result Count $hits";
}
}
```

```

echo '<dl>';
for ($p = 1; $p <= 10; $p++) {
    $rec = yaz_record($id[$i], $p, "string");
    if (empty($rec)) continue;
    echo "<dt><b>$p</b></dt><dd>";
    echo ereg_replace("\n", "<br>\n", $rec);
    echo "</dd>";
}
echo '</dl>';
}
}

```

[10.75.1 yaz_addinfo](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_addinfo](#)(int *id*)

Returns additional error message for target (last request). An empty string is returned if last operation was a success or if no additional information was provided by the target.

[10.75.2 yaz_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_close](#)(int *id*)

Closes a connection to a target. The application can no longer refer to the target with the given id.

[10.75.3 yaz_connect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_connect](#)(string *zurl*)

[yaz_connect\(\)](#) prepares for a connection to a Z39.50 target. The *zurl* argument takes the form `host[:port]/[database]`. If port is omitted 210 is used. If database is omitted Default is used. This function is non-blocking and doesn't attempt to establish a socket – it merely prepares a connect to be performed later when [yaz_wait\(\)](#) is called.

[10.75.4 yaz_errno](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_errno](#)(int *id*)

Returns error for target (last request). A positive value is returned if the target returned a diagnostic code; a value of zero is returned if no errors occurred (success); negative value is returned for other errors targets didn't indicate the error in question.

[yaz_errno\(\)](#) should be called after network activity for each target – (after [yaz_wait\(\)](#) returns) to determine the success or failure of the last operation (e.g. search).

[10.75.5 yaz_error](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_error](#)(int *id*)

Returns error message for target (last request). An empty string is returned if last operation was a success. [yaz_error\(\)](#) returns a english message corresponding to the last error number as returned by [yaz_errno\(\)](#).

[10.75.6 yaz_hits](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_hits](#)(int *id*)

[yaz_hits\(\)](#) returns number of hits for last search.

[10.75.7 yaz_range](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_range](#)(int *id*, int *start*, int *number*)

This function is used in conjunction with [yaz_search\(\)](#) to specify the maximum number of records to retrieve (number) and the first record position (start). If this function is not invoked (only [yaz_search\(\)](#)) start is set to 1 and number is set to 10.

Returns true on success; false on error.

[10.75.8 yaz_record](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_record](#)(int *id*, int *pos*, string *type*)

Returns record at position or empty string if no record exists at given position.

The [yaz_record\(\)](#) function inspects a record in the current result set at the position specified. If no database record exists at the given position an empty string is returned. The argument, type, specifies the form of the returned record. If type is "string" the record is returned in a string representation suitable for printing (for XML and SUTRS). If type is "array" the record is returned as an array representation (for structured records).

[10.75.9 yaz_search](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_search](#)(int *id*, string *type*, string *query*)

[yaz_search\(\)](#) prepares for a search on the target with given id. The type represents the query type – only "rpn" is supported now in which case the third argument is a prefix notation query as used by YAZ. Like [yaz_connect\(\)](#) this function is non-blocking and only prepares for a search to be executed later when [yaz_wait\(\)](#) is called.

[10.75.10 yaz_syntax](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_syntax](#)(int *id*, string *syntax*)

This function is used in conjunction with [yaz_search\(\)](#) to specify the preferred record syntax for retrieval.

[10.75.11 yaz_wait](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_wait](#)(int *id*, string *syntax*)

This function carries out networked (blocked) activity for outstanding requests which have been prepared by the functions [yaz_connect\(\)](#), [yaz_search\(\)](#). [yaz_wait\(\)](#) returns when all targets have either completed all requests or otherwise completed (in case of errors).

[10.76 Compression](#)

[\[Notes en ligne\]](#)

Ce module utilise les fonctions de la librairie zlib ([zlib](#)) de Jean-loup Gailly et Mark Adler pour lire et écrire, de manière transparente, des fichiers compressés avec gzip (.gz). Il faut utiliser la librairie zlib, de version >= 1.0.9.

Ce module contient des versions de la plus part des fonctions du chapitre [10.23 Système de fichiers](#). Mais celles-ci fonctionnent non seulement avec des fichiers compressés, mais aussi des fichiers décompressés (hormis les fonctions utilisant les sockets).

[10.76.1 Petit exemple](#)

[\[Notes en ligne\]](#)

Ouvre un fichier temporaire, écrit un texte et puis affiche deux fois le contenu.

Petit exemple

```

<?php
    $filename = tempnam('/tmp', 'zlibtest').'.gz';
    print "<html>\n<head></head>\n<body>\n<pre>\n";
    $s = "Only a test, test, test, test, test, test, test!\n";
    // ouvre un fichier en écriture, avec compression maximale
    $zp = gzopen($filename, "w9");
    // écrit la chaîne dans le fichier
    gzwrite($zp, $s);
    // ferme le fichier
    gzclose($zp);
    // ouvre en lecture
    $zp = gzopen($filename, "r");
    // lis 3 caractères
    print gzread($zp, 3);
    // Affiche le reste du fichier
    gzpassthru($zp);
    print "\n";

```

```
// ouvre le fichier, et affiche le contenu (deuxième passe)
if (readgzfile($filename) != strlen($s)) {
echo "Error with zlib functions!";
}
unlink($filename);
print "<pre>\n</hl></body>\n</html>\n" ;
?>
```

[10.76.2 gzclose](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzclose](#)(int *zp*)

[gzclose\(\)](#) ferme le pointeur *zp*.

Retourne TRUE ou FALSE, suivant le succès ou l'échec.

Le pointeur de fichier compressé doit être valide, et doit référencer un fichier qui a été ouvert par [gzopen\(\)](#).

[10.76.3 gzeof](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzeof](#)(int *zp*)

Retourne TRUE si le pointeur interne du fichier compressé est à la fin du fichier, sinon retourne FALSE.

Le pointeur de fichier compressé doit être valide, et doit référencer un fichier qui a été ouvert par [gzopen\(\)](#).

[10.76.4 gzfile](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [gzfile](#)(string *filename*, int *use_include_path*)

Identique à [readgzfile\(\)](#), mais [gzfile\(\)](#) retourne un tableau.

Vous pouvez aussi utiliser le deuxième paramètre optionnel en le mettant à "1", si vous voulez rechercher le fichier dans le dossier [7.1.1.13 ini.include_path](#).

Voir aussi [readgzfile\(\)](#), et [gzopen\(\)](#).

[10.76.5 gzgetc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gzgetc](#)(int *zp*)

Retourne une chaîne décompressée, qui contient un caractère unique, lu depuis le fichier référencé par le pointeur *zp*. Retourne FALSE à la fin du fichier (voir [gzeof\(\)](#)).

Le pointeur de fichier compressé doit être valide, et doit référencer un fichier qui a été ouvert par [gzopen\(\)](#).

Voir aussi [gzopen\(\)](#), et [gzgets\(\)](#).

[10.76.6 gzgets](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gzgets](#)(int *zp*, int *length*)

Retourne une chaîne décompressée, de longueur inférieure ou égale à *length* – 1 octets, lue depuis de fichier référencé par le pointeur de fichier *fp*. La lecture s'interrompt lorsque *length* – 1 octets ont été lus, ou bien lorsqu'une nouvelle ligne a été rencontrée, ou bien lorsque la fin du fichier a été atteinte.

Si une erreur survient, retourne *false*.

Le pointeur de fichier compressé doit être valide, et doit référencer un fichier qui a été ouvert par [gzopen\(\)](#).

Voir aussi [gzopen\(\)](#), [gzgetc\(\)](#), et [fgets\(\)](#).

[10.76.7 gzgetss](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gzgetss](#)(int *zp*, int *length*, string *allowable_tags*)

Identique à [gzgets\(\)](#), mais [gzgetss](#) essaie de supprimer toutes les balises HTML et PHP du texte lu.

Vous pouvez utiliser le troisième argument optionnel pour indiquer les balises qui ne doivent pas être supprimées. Note : *allowable_tags* a été ajouté dans PHP 3.0.13, PHP4B3.

Voir aussi [gzgets\(\)](#), [gzopen\(\)](#), et [strip_tags\(\)](#).

[10.76.8 gzopen](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzopen](#)(string *filename*, string *mode*, int *use_include_path*)

Ouvre un fichier compressé avec gzip (.gz) pour le lire ou l'écrire. Le paramètre de mode est le même que dans [fopen\(\)](#) ("rb" ou "wb") mais il peut aussi inclure un niveau de compression ("wb9") ou une stratégie: 'f' pour les données filtrées, comme dans "wb6f", 'h' pour Huffman seul, comme dans "wb1h". (Voir la description de deflateInit2 dans zlib.h pour plus de détails à propos des paramètres de stratégie).

Gzopen peut être utilisé pour ouvrir des fichiers qui ne sont pas au format gzip; dans ce cas, [gzread\(\)](#) lira directement le fichier, sans appliquer de décompression.

Gzopen retourne un pointeur de fichier sur le fichier ouvert. Ce pointeur sera nécessaire pour toutes les opérations ultérieures sur ce fichier. Les opérations de compression/décompression seront transparentes.

Si l'ouverture échoue, la fonction retourne FALSE.

You can use the optional third parameter and set it to "1", if you want to search for the file in the [7.1.1.13 ini.include-path](#), too.

Exemple gzopen()

```
$fp = gzopen("/tmp/file.gz", "r");
```

Voir aussi [gzclose\(\)](#).

[10.76.9 gzpassthru](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzpassthru](#)(int *zp*)

Lit toutes les informations d'un fichier compressé jusqu'à EOF et écrit le résultat décompressé dans la sortie standard.

Si une erreur survient, retourne FALSE.

Le pointeur de fichier doit être valide, et avoir été ouvert par [gzopen\(\)](#).

Le fichier pointé est refermé par [gzpassthru\(\)](#) ce qui le rends inutilisable pour les opérations ultérieures.

[10.76.10 gzputs](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzputs](#)(int *zp*, string *str*, int *length*)

[gzputs\(\)](#) est un alias de [gzwrite\(\)](#), et lui est identique en tous points.

[10.76.11 gzread](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gzread](#)(int *zp*, int *length*)

[gzread\(\)](#) lit jusqu'à *length* octets depuis le fichier compressé référencé par *zp*. La lecture stoppe lorsque *length* octets décompressés ont été lus, ou que la fin du fichier a été trouvée.

```
// lis le contenu d'un fichier compressé et le met dans une chaîne
$filename = "/usr/local/something.txt.gz";
$zd = gzopen( $filename, "r" );
$content = gzread( $zd, 10000 );
gzclose( $zd );
```

Voir aussi [gzwrite\(\)](#), [gzopen\(\)](#), [gzgets\(\)](#), [gzgetss\(\)](#), [gzfile\(\)](#), et [gzpassthru\(\)](#).

[10.76.12 gzrewind](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzrewind](#)(int *zp*)

Positionne le pointeur interne du fichier au début du fichier compressé.

Si une erreur survient, retourne 0.

Le pointeur de fichier doit être valide, et avoir été retourné par la fonction [gzopen\(\)](#).

Voir aussi [gzseek\(\)](#) et [gztell\(\)](#).

[10.76.13 gzseek](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzseek](#)(int *zp*, int *offset*)

Positionne le pointeur interne du fichier compressé *zp* à la position donnée en *offset*. Equivalent à l'appel (en C) de `gzseek(zp, offset, SEEK_SET)`.

Si le fichier est ouvert en lecture seule, cette fonction émulée peut être extrêmement lente. Si le fichier est ouvert en lecture, seul le déplacement avant (forward seek) sont acceptés. `gzseek` compresse alors une

séquence de zéro jusqu'à la nouvelle position.

Retourne 0 en cas de succès, sinon retourne -1. Notez positionner le pointeur au delà de la fin du fichier est une erreur.

Voir aussi [gztell\(\)](#) et [gzrewind\(\)](#).

10.76.14 gztell

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gztell](#)(int *zp*)

Retourne la position du pointeur interne du fichier référencé par *zp*, i.e., son offset en octets depuis le début du fichier.

Si une erreur survient, retourne FALSE.

Le pointeur de fichier doit être valide, et doit avoir été retourné par la fonction [gzopen\(\)](#).

Voir aussi [gzopen\(\)](#), [gzseek\(\)](#) et [gzrewind\(\)](#).

10.76.15 gzwrite

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzwrite](#)(int *zp*, string *string*, int *length*)

[gzwrite\(\)](#) écrit le contenu de la chaîne *string* dans le fichier compressé référencé par le pointeur *zp*. Si l'argument *length* est donné, l'écriture cessera après avoir écrit *length* octets (non compressé), ou bien si la fin de la chaîne a été atteinte.

Notez que si l'argument *length* est donnée, alors l'option [7.1.1.17 ini.magic-quotes-runtime](#) sera ignorée et les slashes ne seront pas supprimés de la chaîne *string*.

Voir aussi [gzread\(\)](#), [gzopen\(\)](#), et [gzputs\(\)](#).

10.76.16 readgzfile

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [readgzfile](#)(string *filename*, int *use_include_path*)

Lit un fichier, le décompresse et l'écrit dans la sortie standard.

[Readgzfile\(\)](#) peut être utilisé pour lire un fichier qui n'est pas au format gzip, car dans ce cas, la décompression est omise, et le fichier est directement affiché.

Retourne le nombre d'octets (non compressé) lus depuis le fichier. Si une erreur survient, retourne FALSE, et à moins que la fonction n'ait été appelée avec `@readgzfile`, l'erreur est affichée.

Le fichier *filename* sera ouvert et son contenu sera écrit dans la sortie standard.

Vous pouvez utiliser le paramètre optionnel en le mettant à "1", si vous voulez rechercher le fichier dans le dossier [7.1.1.13 ini.include-path](#).

Voir aussi [gzpassthru\(\)](#), [gzfile\(\)](#), et [gzopen\(\)](#).

A Débugueur PHP

[\[Notes en ligne\]](#)

A.1 Utiliser le débugeur PHP

[\[Notes en ligne\]](#)

Le débugeur PHP sert à repérer les bugs récalcitrants. Le débugeur fonctionne en se connectant à un port *TCP* à chaque démarrage de PHP. Tous les messages d'erreur seront envoyés sur cette connexion. Cette page est faite pour un "serveur de débugeage", qui peut peut fonctionner avec un *IDE* ou un éditeur programmable (tel que Emacs).

Comment paramétrer le débugeur :

1. Réservez un port TCP pour le débugeur dans le fichier [7.1 Le fichier de configuration](#) ([7.1.4.2 ini.debugger.port](#)) et activez le ([7.1.4.3 ini.debugger.enabled](#)).
2. Configurer un client TCP sur ce port (par exemple `socket -l -s 1400` sous UNIX).
3. Dans votre code, placez la ligne "`debugger_on(host)`", où *host* est l'IP ou le nom de l'hôte qui supporte le client *TCP*.

Désormais, toutes les alertes, notes, ... seront envoyées sur la socket client, *même si vous avez inactivé le rapport d'erreur avec [error_reporting\(\)](#)*.

Note : *Le code du débugeur n'a pas été porté en PHP 4.0, au moment de la rédaction de ce manuel : seule la version 3.0 accepte le débugeur.*

A.2 Protocole du débugeur

[\[Notes en ligne\]](#)

Le protocole de débugeage est ligne par ligne. Chaque ligne a un type *type*, et plusieurs lignes composent un message. Chaque message commence avec une ligne du type *start* et se termine avec une ligne de type *end*. PHP peut envoyer des lignes de plusieurs messages simultanément.

Voici un exemple de ligne à ce format :

```
date time
host(pid)
type:
message-data
```

date

- Les dates sont au format ISO 8601 (*yyyy-mm-dd*)

time

- Les heures, y compris les micro-secondes: *hh:mm:uuuuuuu*

host

- Le nom DNS ou adresse IP de l'hôte qui a généré l'erreur.

pid

- PID (process id) sur l'hôte *host*, qui a généré l'erreur.

type

- Type de la ligne. Indique au programme client comment traiter les données suivantes :

Nom	Signification
start	Indique au programme client que le message du débogueur commence ici. Le contenu de <i>data</i> sera un type d'erreur, comme listé ci dessous. @tab
message	Le message d'erreur PHP.
location	Nom du fichier, et numéro de ligne, où l'erreur est survenue. La première occurrence de location contiendra toujours la localisation générale. <i>data</i> contiendra : <i>file:line</i> . Il y a toujours une indication de location après un message et après chaque fonction. @tab
frames	Nombre de frames dans le dump de la pile. Si il y a 4 frames, attendez vous à des information sur 4 niveaux de fonctions. Si la ligne "frame" n'existe pas, la profondeur doit être 0 (une erreur est survenue au niveau général). @tab
function	Nom de la fonction qui a généré l'erreur. Elle sera répétée à chaque niveau de la pile d'appel. @tab
end	Indique au client que le message du débogueur se termine ici. @tab

data

- Ligne de données.

Débogueur	Interne PHP
alerte (warning)	E_WARNING
erreur	E_ERROR
analyse (parse)	E_PARSE
note (notice)	E_NOTICE
core-error	E_CORE_ERROR
core-warning	E_CORE_WARNING
inconnue	(toutes les autres)

Exemple de message du débogueur

```
1998-04-05 23:27:400966 lucifer.guardian.no(20481) start: notice
1998-04-05 23:27:400966 lucifer.guardian.no(20481) message: Uninitialized variable
1998-04-05 23:27:400966 lucifer.guardian.no(20481) location: (null):7
```

PHP 4.0 Manuel de Référence.

```
1998-04-05 23:27:400966 lucifer.guardian.no(20481) frames: 1
1998-04-05 23:27:400966 lucifer.guardian.no(20481) function: display
1998-04-05 23:27:400966 lucifer.guardian.no(20481) location: /home/ssb/public_html/test.php3:10
1998-04-05 23:27:400966 lucifer.guardian.no(20481) end: notice
```

[B Migration de PHP/FI 2.0 à PHP 3.0](#)

[\[Notes en ligne\]](#)

[B.1 A propos des incompatibilités en 3.0](#)

[\[Notes en ligne\]](#)

PHP 3.0 a été entièrement réécrit. Le nouvel analyseur syntaxique est beaucoup plus robuste et cohérent qu'en version 2.0. Il est aussi nettement plus rapide, et utilise encore moins de mémoire. Cependant, ces améliorations n'ont pu être possible qu'au prix de modifications parfois importantes, tant au niveau des syntaxes, qu'au niveau des fonctionnalités.

De plus, l'équipe de développement PHP a essayé de nettoyer la syntaxe et les sémantiques, ce qui a aussi causé quelques incompatibilités. A long terme, nous pensons que ces modifications seront pour le bien de tous.

Ce chapitre va tenter de vous montrer les incompatibilités que vous pourriez rencontrer lors de votre migration de PHP/FI 2.0 à PHP 3.0 et de vous aider à les résoudre. Les nouvelles fonctionnalités ne sont pas signalées, à moins que cela ne soit nécessaire.

Un programme de conversion automatique de vos vieux script PHP/FI 2.0 existe. Il est disponible dans le dossier de convertisseur de la distribution PHP 3.0. Ce programme ne fait que repérer les modifications de syntaxe, et ne vous épargnera pas une relecture attentive du script.

[B.2 Start/end tags](#)

[\[Notes en ligne\]](#)

La première chose que vous remarquerez probablement est que les balises de PHP start et end ont changé. L'ancienne forme `<? >` a été remplacée par trois nouvelles balises possibles : ***Migration: Migration: balises start/end***

```
<? echo "Ceci est du code PHP/FI 2.0.\n"; >
```

Comme en version 2.0, PHP/FI accepte aussi cette variante : ***Migration: premières nouvelles balises start/end***

```
<? echo "Ceci est du code PHP 3.0!\n"; ?>
```

Notez bien que la balise de fin contient désormais un point d'interrogation et un signe " supérieur à " `>`. Cependant, si vous souhaitez utiliser XML sur votre serveur, vous aurez sûrement des problèmes avec cette variante, car PHP risque d'essayer d'exécuter des balises XML. A cause de ceci, la notation suivante a été ajoutée : ***Migration: second new start/end tags***

```
<?php echo "This is PHP 3.0 code!\n"; ?>
```

Some people have had problems with editors that don't understand the processing instruction tags at all.

Microsoft FrontPage is one such editor, and as a workaround for these, the following variation was introduced as well: *Deuxièmes nouvelles balises start/end*

```
<script language="php">
echo "Ceci est du code PHP 3.0!\n";
</script>
```

B.3 if..endif syntax

[\[Notes en ligne\]](#)

La syntaxe alternative pour écrire des instructions if/elseif/else, avec if(); elseif(); else; endif; ne pouvait pas être conservée sans ajouter beaucoup de complexité à l'analyseur syntaxique. De ce fait, cette syntaxe à changée : *Migration: ancienne syntaxe if..endif*

```
if ($foo);
echo "oui\n";
elseif ($bar);
echo "presque\n";
else;
echo "non\n";
endif;
```

Migration: nouvelle syntaxe if..endif

```
if ($foo):
echo "oui\n";
elseif ($bar):
echo "presque\n";
else:
echo "non\n";
endif;
```

Notez que les points virgules ont été remplacée par des point dans toutes les commandes, sauf pour la dernière expression (endif).

B.4 while syntax

[\[Notes en ligne\]](#)

Tout comme pour if..endif, la syntaxe des boucles while..endwhile a changée as well: *Migration: ancienne syntaxe while..endwhile*

```
while ($more_to_come);
...
```

```
endwhile;
```

Migration: nouvelle syntaxe while..endwhile

```
while ($more_to_come):
    ...
endwhile;
```

Attention Si vous utilisez la vieille syntaxe while..endwhile en PHP 3.0, vous obtiendrez une boucle sans fin !

B.5 Types d'expression

[\[Notes en ligne\]](#)

PHP/FI 2.0 utilisait le membre à gauche dans les expressions, pour déterminer le type de résultat attendu. PHP 3.0 prend en compte les deux cotés de l'expression, et cela peut produire des résultats inattendus avec les scripts 2.0.

Considérez les lignes suivantes:

```
$a[0]=5;
$a[1]=7;
$key = key($a);
while ("" != $key) {
echo "$keyn";
next($a);
}
```

En PHP/FI 2.0, cet exemple va afficher les indices des \$a. En PHP 3.0, l'exemple ne va rien afficher du tout. La raison est qu'en PHP 2.0, puisque l'argument de gauche est de type chaîne, une comparaison de chaîne était effectué, et, effectivement, "" n'est pas "", ce qui conduit la boucle à continuer. En PHP 3, lorsqu'une chaîne est comparée avec un entier, la comparaison est de type chaîne (la chaîne est convertie en entier). Ce qui revient à faire la comparaison entre (atoi("")) qui vaut 0, et la variablelist qui vaut aussi 0, et comme 0==0, la boucle ne commence même pas.

La correction de ceci est simple : il suffit de remplacer les commandes while par:

```
while ((string)$key != "") {
```

B.6 Les messages d'erreur ont changé

[\[Notes en ligne\]](#)

Les messages d'erreur en PHP 3.0 sont généralement plus précis que ceux de la version 2.0., mais vous ne verrez plus la portion de code qui a causé l'erreur. A la place, un numéro de ligne et un nom de fichier sera retourné.

[B.7 Evaluation rapide des booléens](#)

[\[Notes en ligne\]](#)

En PHP 3., l'évaluation des est court circuité. Cela signifie dans une expression telle que ((1 || test_me())), la fonction @xref{function.test-me , , test_me()} ne sera pas exécutée, car cela ne changera pas le résultat. C'est une amélioration mineure, mais qui peut avoir des effets secondaires importants.

[B.8 La valeur true/false comme retour de fonctions](#)

[\[Notes en ligne\]](#)

La plupart des fonctions internes de PHP ont été réécrite pour qu'elle retourne TRUE en cas de succès, et FALSE en cas d'erreur, au contraire des fonctions qui retournaient 0 et -1 en PHP/FI 2.0. Le nouveau comportement est beaucoup plus logique, comme par exemple \$fp = fopen("/your/file") or fail("fichier non trouvé!");. Etant donné que PHP/FI 2.0 n'a pas de règle claire à propos de ce que les fonctions doivent retourner en cas d'échec, la plus part des scripts devront probablement être vérifié manuellement, après avoir utilisé le convertisseur 2.0 à 3.0.

Migration depuis 2.0: valeur retournées, ancienne façon

```
$fp = fopen($file, "r");
if ($fp == -1);
echo("Impossible d'ouvrir le fichier $file en lecture <br>\n");
endif;
```

Migration depuis 2.0: valeur retournées, nouvelle façon

```
$fp = @fopen($file, "r") or print("Impossible d'ouvrir le fichier $file en lecture<br>\n");
```

[B.9 Diverses incompatibilités](#)

[\[Notes en ligne\]](#)

- Le module PHP 3.0 pour Apache n'accepte plus les versions d'Apache antérieure à la version 1.2. Apache 1.2 ou plus récent est nécessaire.
- [echo\(\)](#) n'utilise plus de chaîne de formatage. Il faut utiliser [printf\(\)](#) à la place.
- En PHP/FI 2.0, un effet secondaire de l'implémentation faisait que \$foo[0] était la même chose que \$foo. Ce n'est plus vrai en PHP 3.0.
- Lire un tableau avec \$array[] n'est plus valable. Ainsi, il n'est plus possible de passer en revue un tableau avec des boucles telles que \$data = \$array[]. Utilisez [current\(\)](#) et [next\(\)](#) à la place. Ainsi, \$array1[] = \$array2 n'ajoute pas les valeurs de \$array2 à \$array1, mais crée un nouvel élément dans \$array1 et y affecte \$array2 comme dernier élément. Voir aussi les tableaux multidimensionnels.
- "+" n'est plus utilisable comme opérateur de concaténation de chaîne. A la place, il converti les arguments en nombres, et effectue une addition numérique. Utilisez "." à la place.

Migration depuis 2.0: concaténation de chaînes

```
echo "1" + "1";
```

En PHP 2.0 cela retournerait 11, en PHP 3.0 cela va retourner 2. A la place, faites :

```
echo "1"."1";
```

```
$a = 1;  
$b = 1;  
echo $a + $b;
```

Cela va afficher 2, tant en PHP 2.0 qu'en 3.0.

```
$a = 1;  
$b = 1;  
echo $a.$b;
```

Cela va afficher 11 en PHP 3.0.

C Développement PHP

[\[Notes en ligne\]](#)

C.1 Adding functions to PHP3

[\[Notes en ligne\]](#)

C.1.1 Prototypes de fonctions

[\[Notes en ligne\]](#)

Toutes les fonctions suivent le schéma suivant :

```
void php3_foo(INTERNAL_FUNCTION_PARAMETERS) {  
}
```

Même si votre fonction ne prend aucun argument, c'est comme cela qu'elle doit être appelée.

C.1.2 Arguments de fonctions

[\[Notes en ligne\]](#)

Les arguments sont toujours de type val. Ce type contient un membre de type union, qui indique le type réel d'argument. De cette façon, si votre fonction prend deux arguments, elle ressemble à ceci :

Argument de fonction de lecture

```
pval *arg1, *arg2;  
if (ARG_COUNT(ht) != 2 || getParameters(ht,2,&arg1,&arg2)==FAILURE) {  
    WRONG_PARAM_COUNT;  
}
```

NOTE: Les arguments peuvent être passés par valeur ou par référence. Dans les deux cas, vous devez passer `&(pval *)` à `getParameters`. Si vous voulez vérifier que le n-ième paramètre a été passé par référence ou par valeur, vous devez utiliser la fonction `ParameterPassedByReference(ht,n)`. Elle retournera 1 ou 0. Lorsque vous modifiez l'un des paramètres, qu'ils soient envoyés par référence ou par valeur, vous pouvez le passer à `pval_destructor` pour le réinitialiser, ou, s'il s'agit d'un tableau et que vous voulez ajouter des valeurs, vous pouvez utiliser des fonctions similaires à celles qui sont dans `internal_functions.h`, qui manipule `return_value` comme tableau.

Par ailleurs, si vous modifiez un paramètre en `IS_STRING`, assurez-vous que vous avez bien assigné une nouvelle chaîne avec `@xref{function.estrdup, , estrdup()}` et une nouvelle longueur de chaîne. Seulement après, vous pouvez modifier le type en `IS_STRING`. Si vous modifiez une chaîne en `IS_STRING` ou `IS_ARRAY` vous devez d'abord appeler le destructeur `pval_destructor`.

C.1.3 Fonctions à nombre d'arguments variable

[\[Notes en ligne\]](#)

Une fonction peut prendre un nombre variable d'arguments. Si votre fonction peut prendre deux ou trois arguments, utiliser la syntaxe suivante :

Fonctions à nombre d'arguments variable

```

pval *arg1, *arg2, *arg3;
int arg_count = ARG_COUNT(ht);
if (arg_count < 2 || arg_count > 3 ||
    getParameters(ht, arg_count, &arg1, &arg2, &arg3) == FAILURE) {
    WRONG_PARAM_COUNT;
}

```

C.1.4 Utiliser les arguments d'une fonction

[\[Notes en ligne\]](#)

De type de chaque argument est stocké dans le champs pval. Ce champs peut prendre les valeurs suivantes :

IS_STRING	Chaîne de caractères
IS_DOUBLE	Nombre à virgule flottante, en précision double
IS_LONG	Entier long
IS_ARRAY	Tableau
IS_EMPTY	Aucune
IS_USER_FUNCTION	??
IS_INTERNAL_FUNCTION	?? (Si ce type ne peut pas être passé à une fonction, effacez le)
IS_CLASS	??
IS_OBJECT	??

Si vous recevez un argument d'un type, te que vous voulez l'utiliser avec un autre type, ou si vous voulez simplement forcer le type, vous pouvez utiliser l'une des fonctions de conversion suivantes :

```

convert_to_long(arg1);
convert_to_double(arg1);
convert_to_string(arg1);
convert_to_boolean_long(arg1); /* Si la chaîne est "" ou "0" elle devient 0, 1 sinon */
convert_string_to_number(arg1); /* Converti une chaîne en LONG ou DOUBLE suivant la chaîne */

```

Ces fonctions convertissent sur place : elles ne retourne aucune valeur.

La valeur de l'argument est enregistrées dans une union. Les membres sont :

- IS_STRING: arg1->value.str.val
- IS_LONG: arg1->value.lval
- IS_DOUBLE: arg1->value.dval

C.1.5 Gestion de la mémoire dans une fonction

[\[Notes en ligne\]](#)

Toute la mémoire nécessaire à une fonction doit être allouée avec `emalloc()` ou `estrdup()`. Ces fonctions ont le goût et l'odeur des classiques `malloc()` et `strdup()`. La mémoire doit être libérée avec `efree()`.

Il y a deux types de mémoire dans ce programme : la mémoire qui est retournée à l'analyseur, et la mémoire qui est nécessaire pour le stockage temporaire dans la fonction. Lorsque vous assignez une chaîne dans une variable qui est retournée à l'analyseur, assurez-vous de bien allouer la mémoire avec `emalloc()` ou `estrdup()`. Cette mémoire ne doit JAMAIS être libérée, sauf si vous réécrivez votre original plus loin, dans la même fonction (mais ce n'est pas de la programmation propre).

Pour tous vos besoins en mémoire temporaire/permanente dont vous avez besoin dans vos fonctions/librairies, vous devez utiliser les fonctions `emalloc()`, `estrdup()` et `efree()`. Elles se comportent EXACTEMENT comme leurs homologues. Tout ce qui est créé avec `emalloc()` ou `estrdup()` doit être libéré avec `efree()` à un moment ou un autre, à moins que ce ne soit utile ailleurs dans le programme; sinon, il va y avoir une fuite de mémoire. La signification de "Elles se comportent EXACTEMENT comme leurs homologues" est que si vous libérez une variable qui n'a pas été créée avec `emalloc()` ou `estrdup()`, vous courez droit à un crash ("segmentation fault"). Soyez alors extrêmement prudent, et libérez toute votre mémoire inutilisée.

Si vous compilez avec `-DDEBUG`, PHP 3 affichera la liste de tous les appels à `emalloc()` et `estrdup()` mais jamais à `efree()` lorsque celui-ci intervient dans un script spécifié.

C.1.6 Affecter une variable dans la table des symboles

[\[Notes en ligne\]](#)

Un grand nombre de macros sont disponibles pour rendre plus facile l'insertion de variables dans la table des symboles :

- `SET_VAR_STRING(name,value)`
- `SET_VAR_DOUBLE(name,value)`
- `SET_VAR_LONG(name,value)`

Soyez prudent. La valeur doit être placée dans une portion de mémoire créée avec `malloc()`, sinon le gestionnaire de mémoire essaiera de libérer le pointeur plus tard. Ne passez aucune mémoire allouée statiquement à `SET_VAR_STRING`.

Les tables des symboles de PHP 3 est une table de hash. A n'importe quel moment, `&symbol_table` est un pointeur sur la table principale, et `active_symbol_table` pointe sur la table actuellement utilisée. (ces deux tables peuvent être identiques au démarrage, ou différent, suivant que vous êtes dans une fonction ou non). Les exemples suivants utilisent `'active_symbol_table'`. Vous devriez la remplacer par `&symbol_table` si vous voulez travailler sur la table principale. De plus, les mêmes fonctions peuvent être appliquées à des tableaux, comme expliqué ci dessous.

Vérification de l'existence de \$foo dans la table des symboles

```
if (hash_exists(active_symbol_table,"foo",sizeof("foo"))) { existe... }
else { n'existe pas }
```

Rechercher la taille d'une variable dans la table des symboles

```
hash_find(active_symbol_table, "foo", sizeof("foo"), &pvalue);
check(pvalue.type);
```

En PHP 3.0, les tableaux sont implémentés en utilisant les mêmes tables de hash que les variables. Cela signifie que les deux fonctions ci dessus peuvent être appelées pour vérifier la présence de variables dans un tableau.

Si vous voulez définir un nouveau tableau dans la table des symboles, utilisez le code suivant.

D'abord, vous devez vérifier qu'il n'existe pas, avec `hash_exists()` ou `hash_find()`.

Puis, initialisez le tableau :

Initialisation d'un tableau

```
pval arr;
if (array_init(&arr) == FAILURE) { failed... };
hash_update(active_symbol_table, "foo", sizeof("foo"), &arr, sizeof(pval), NULL);
```

Ce code déclare un nouveau tableau, appelé \$foo, dans la table de symbole Ce tableau est vide.

Voici comment ajouter deux nouvelles entrées dans ce tableau :

Ajout d'entrées dans un tableau.

```
pval entry;
entry.type = IS_LONG;
entry.value.lval = 5;
/* définit $foo["bar"] = 5 */
hash_update(arr.value.ht, "bar", sizeof("bar"), &entry, sizeof(pval), NULL);
/* définit $foo[7] = 5 */
hash_index_update(arr.value.ht, 7, &entry, sizeof(pval), NULL);
/* définit la prochaine place libre dans $foo[],
 * $foo[8], qui sera 5 (comme en php2)
 */
hash_next_index_insert(arr.value.ht, &entry, sizeof(pval), NULL);
```

Si vous voulez modifier une valeur que vous avez inséré dans une table de hash, vous devez d'abord la lire dans la table. Pour éviter cette recherche, vous pouvez fournir une pval ** à la fonction d'ajout dans la table de hash, et elle modifiera la valeur à l'adresse pval *, avec la valeur donnée. Si cette valeur est NULL, (comme dans tous les exemples ci dessus), ce paramètre sera ignoré.

`hash_next_index_insert()` utiliser plus ou moins la même logique que "\$foo[] = bar;" in PHP 2.0.

Si vous construisez un tableau, pour le retourner, vous pouvez l'initialiser comme ceci :

```
if (array_init(return_value) == FAILURE) { échec...; }
```

...puis ajouter les valeurs grâce aux macros:

```
add_next_index_long(return_value, long_value);
add_next_index_double(return_value, double_value);
add_next_index_string(return_value, estrdup(string_value));
```

Bien sur, si l'ajout n'est pas fait juste après l'initialisation, vous devrez d'abord rechercher le tableau :

```
pval *arr;
```

```
if (hash_find(active_symbol_table,"foo",sizeof("foo"),(void **)&arr)==FAILURE) { introuvable... }
else { utilisez arr->value.ht... }
```

Notez que `hash_find` reçoit un pointeur sur un pointeur sur `pval`, et pas un pointeur sur `pval`. Toutes les fonctions d'accès aux hash retourne SUCCES (SUCCESS) ou ECHEC (FAILURE), excepté `hash_exists()`, qui retourne un booléen.

C.1.7 Retourne une valeur simple

[\[Notes en ligne\]](#)

Un grand nombre de macros sont disponible pour simplifier le retour des valeurs. La macro `RETURN_*` fixe la valeur de retour, et termine la fonction :

- `RETURN`
- `RETURN_FALSE`
- `RETURN_TRUE`
- `RETURN_LONG(l)`
- `RETURN_STRING(s,dup)` Si `dup` est `TRUE`, duplique la chaîne.
- `RETURN_STRINGL(s,l,dup)` Retourne la chaîne (`s`) en spécifiant la longueur (`l`).
- `RETURN_DOUBLE(d)`

La macro `RETVAL_*` macros fixe la valeur de retour, mais ne termine pas la fonction.

- `RETVAL_FALSE`
- `RETVAL_TRUE`
- `RETVAL_LONG(l)`
- `RETVAL_STRING(s,dup)` Si `dup` est `TRUE`, duplique la chaîne
- `RETVAL_STRINGL(s,l,dup)` Retourne la chaîne (`s`) en spécifiant la longueur (`l`).
- `RETVAL_DOUBLE(d)`

Les macros ci dessus vont utiliser `estrdup()` sur les arguments passés. Cela vous permet de libérer tranquillement les arguments après avoir appelé cette fonction, ou bien, utiliser de la mémoire allouée statiquement.

Si votre fonction retourne un booléen de succès/erreur, utilisez toujours `RETURN_TRUE` et `RETURN_FALSE` respectivement.

C.1.8 Retourner des valeurs complexes

[\[Notes en ligne\]](#)

Votre fonction peut aussi retourner des valeurs complexes, tels que des objets ou tableaux. Retourner un objet:

1. Appeler `object_init(return_value)`.
2. Remplissez les valeurs. Les fonctions utilisables sont listées ci dessous.
3. Eventuellement, enregistrez les fonctions pour cet objet. Afin de lire des valeurs de cet objet, la fonction doit lire dans "this", dans la table de symbole active `active_symbol_table`. Son type doit être `IS_OBJECT`, et c'est une table de hash basique. (i.e., vous pouvez utiliser les fonctions habituelles de

.value.ht). L'enregistrement réel peut être fait comme suit :

```
add_method( return_value, function_name, function_ptr );
```

Les fonctions d'accès aux objets sont :

- `add_property_long(return_value, property_name, l)` – Ajoute un membre nommé 'property_name', de type long, égal à 'l'
- `add_property_double(return_value, property_name, d)` – Idem, ajoute un double
- `add_property_string(return_value, property_name, str)` – Idem, ajoute une chaîne
- `add_property_stringl(return_value, property_name, str, l)` – Idem, ajoute une chaîne de longueur 'l'.

Retournez un tableau :

1. Appelez `array_init(return_value)`.
2. Remplissez les valeurs. Les fonctions disponibles sont listées ci dessous.

Les fonctions utilisées pour accéder à un tableau sont :

- `add_assoc_long(return_value,key,l)` – Ajoute une entrée associative avec la clé 'key' et la valeur 'l', de type long
- `add_assoc_double(return_value,key,d)` – Ajoute une entrée associative avec la clé 'key' et la valeur 'l', de type double
- `add_assoc_string(return_value,key,str,duplicate)`
- `add_assoc_stringl(return_value,key,str,length,duplicate)` specify the string length
- `add_index_long(return_value,index,l)` – Ajoute une entrée d'index 'index' avec la valeur 'l', de type long
- `add_index_double(return_value,index,d)`
- `add_index_string(return_value,index,str)`
- `add_index_stringl(return_value,index,str,length)` – spécifie la longueur de la chaîne.
- `add_next_index_long(return_value,l)` – ajoute une entrée tableau, dans le prochain offset libre, de longueur 'l', de type long
- `add_next_index_double(return_value,d)`
- `add_next_index_string(return_value,str)`
- `add_next_index_stringl(return_value,str,length)` – specify the string length

C.1.9 Using the resource list

[\[Notes en ligne\]](#)

PHP 3.0 dispose de standard pour traiter un certains nombre de ressources. Ils remplacent tous les listes de PHP 2.0.

Fonctions accessibles :

- `php3_list_insert(ptr, type)` – retourne l'identifiant 'id' de la nouvelle ressource insérée.

- `php3_list_delete(id)` – efface la ressource d'identifiant `id`
- `php3_list_find(id,*type)` – retourne le pointeur de la ressource d'identifiant `id`, et modifie le type '`type`'

Typiquement, ces fonctions sont utilisées pour les pilotes SQL, mais elles peuvent servir n'importe quoi d'autre. Par exemple, conserver un pointeur de fichier.

La liste standard de code ressemble à ceci :

Ajouter une nouvelle ressource

```
RESOURCE *resource;
/* ...alloue de la mémoire pour la ressource, et l'acquiert ... */
/* Ajoute la nouvelle ressource dans la liste */
return_value->value.lval = php3_list_insert((void *) resource, LE_RESOURCE_TYPE);
return_value->type = IS_LONG;
```

Utiliser une ressource existante

```
pval *resource_id;
RESOURCE *resource;
int type;
convert_to_long(resource_id);
resource = php3_list_find(resource_id->value.lval, &type);
if (type != LE_RESOURCE_TYPE) {
php3_error(E_WARNING,"resource index %d has the wrong type",resource_id->value.lval);
RETURN_FALSE;
}
/* ...utiliser la ressource... */
```

Effacer une ressource existante

```
pval *resource_id;
RESOURCE *resource;
int type;
convert_to_long(resource_id);
php3_list_delete(resource_id->value.lval);
```

Les types de ressources doivent être enregistré dans le fichier `php3_list.h`, dans l'énumération `list_entry_type`. En plus, il faut penser à ajouter une fonction de terminaison, pour chaque type de ressource défini, dans le fichier `list.c`, pour la fonction `list_entry_destructor()` (même si vous n'avez rien de particulier à faire lors de la terminaison, vous devez au moins ajouter un cas vide).

C.1.10 Utiliser la table des ressources persistantes.

[\[Notes en ligne\]](#)

PHP 3.0 dispose d'un lieu de stockage des ressources persistantes (i.e., les ressources qui doivent être conservées d'un hit à l'autre). Le premier module à utiliser cette capacité a été MySQL, et mSQL suivi, ce qui fait que l'on peut se faire une impression du fonctionnement de cette fonction avec `mysql.c`. Les fonctions ressemblent à ceci :

- `php3_mysql_do_connect`
- `php3_mysql_connect()`
- `php3_mysql_pconnect()`

L'idée conductrice de ces modules est la suivante :

1. Programmez tout votre module pour qu'il travaille avec les ressources standard, comme mentionné dans la section (9).
2. Ajoutez un autre fonction de connexion, qui vérifie d'abord que la ressource existe dans la liste des ressources persistantes. Si c'est le cas, enregistrez cette ressource comme pour les ressources standard (et gr'ce à la première étape, cela va fonctionner immédiatement). Si la ressource n'existe pas, créez la, ajoutez la à la liste de ressources persistantes, et ajoutez la à la liste de ressource, ce qui fait que le code va fonctionner, et que le prochain appel renverra une ressource existante. Vous devez enregistrer ces fonctions avec un type différent (`LE_MYSQL_LINK` pour les liens non persistants, et `LE_MYSQL_PLINK` pour les liens persistants).

Si vous jetez un oeil dans `mysql.c`, vous verrez que, hormis la fonction de connexion complexe, rien n'a du être changé dans le module.

La même interface existe pour la liste des ressources standard, et pour la liste des ressources persistantes, seule la 'list' est remplacée par 'plist':

- `php3_plist_insert(ptr, type)` – retourne l'identifiant 'id' de la nouvelle ressource insérée.
- `php3_plist_delete(id)` – efface la ressource d'identifiant id
- `php3_plist_find(id,*type)` – retourne le pointeur de la ressource d'identifiant id, et modifie le type 'type'

Cependant, il est probable que ces fonctions seront inutiles pour vous, lorsque vous essayerez d'implémenter un module persistant. Typiquement, on utilise le fait que la liste de ressources persistante est une table de hash. Par exemple, dans les modules MySQL/mSQL, lors d'un appel à `pconnect()`, la fonction construit une chaîne avec l'hôte/utilisateur/mot_de_passe, et l'utilise pour enregistrer dans la table de hash. Au prochain appel, avec les mêmes hôte/utilisateur/mot_de_passe, la même clé sera générée, et la ressource associée sera retrouvée.

Jusqu'à ce que la documentation s'étoffe, jetez un oeil aux fichiers `mysql.c` ou `mSQL.c` pour voir comment implémenter vos accès aux ressources persistantes.

Une chose importante à noter : les ressources qui sont enregistrées dans la liste de ressource persistante ne DOIVENT PAS être allouées avec le gestionnaire de mémoire PHP, c'est à dire qu'elles ne doivent pas être créées avec `emalloc()`, `estrdup()`, etc. Au contraire, il faut utiliser les fonctions standard `malloc()`, `strdup()`, etc. La raison est for simple : à la fin de la requête, la mémoire sera supprimée par le gestionnaire. Etant donné que les liens persistants doivent être conservés, il ne faut pas utiliser le gestionnaire de mémoire.

Lorsque vous enregistrez une ressource qui sera placée dans la liste de ressource persistantes, il faut ajouter les destructeurs dans les deux listes de ressources, persistante ou pas. Le destructeur de la liste de ressources non persistantes ne doit rien faire du tout, tandis que celui de la liste de ressources persistantes doit libérer proprement toutes les ressources acquises (mémoire, lien SQL...). Comme pour les ressources non persistantes vous DEVEZ ajouter un destructeur, même si il ne fait rien. N'oubliez pas que `emalloc()` et compagnie ne doivent pas être utilisés en conjonction avec la liste de ressources persistantes, et donc, vous ne devez pas utiliser `efree()` non plus.

C.1.11 Ajouter des directives de configuration à l'exécution

[\[Notes en ligne\]](#)

De nombreuses caractéristiques de PHP3 peuvent être configurée à l'exécution. Ces directives peuvent apparaître dans le fichier `php3.ini`, ou, dans le cas du module Apache, dans le fichier `.conf`. L'avantage de l'avoir dans le fichier `.conf`, est que ces caractéristiques peuvent être configurées dossier par dossier. Cela signifie qu'un dossier peut avoir un `safe mode exec dir`, tandis qu'un autre en aura un autre. Cette granularité de la configuration peut être extrêmement pratique lorsque le serveur supporte plusieurs serveurs virtuels. Les étapes de configuration d'une nouvelle directive sont :

1. Ajouter la directive à la structure `php3_ini_structure` dans le fichier `mod_php3.h`.
2. Dans `main.c`, éditez la fonction `php3_module_startup` et ajoutez l'appel approprié à `cfg_get_string()` ou `cfg_get_long()`.
3. Ajoutez la directive, ses restrictions et un commentaire dans la structure `php3_commands` du fichier `mod_php3.c`. Notez la partie restrictions. `RSRC_CONF` sont des directives qui ne peuvent être disponibles que dans le fichier de configuration Apache. Toutes les directives `OR_OPTIONS` peuvent être placées n'importe où, y compris dans un fichier `.htaccess`.
4. Soit dans `php3take1handler()`, soit dans `php3flaghandler()`, ajoutez l'entrée appropriée pour votre directive.
5. Dans la section de configuration, de `_php3_info()`, dans le fichier `functions/info.c`, vous devez ajouter votre configuration.
6. Finalement, vous devez utiliser votre configuration quelque part. Elle sera accessible par `php3_ini.directive`.

C.2 Appeler des fonctions utilisateurs

[\[Notes en ligne\]](#)

Pour appeler des fonctions utilisateurs depuis une fonction interne, vous devez utiliser la fonction `@xref{function.call-user-function , , call_user_function()}`.

`@xref{function.call-user-function , , call_user_function()}` retourne `SUCCESS` en cas de succès, et `FAILURE` en cas d'échec, ou si la fonction n'a pas été trouvée. Vous devez vérifier cette valeur. Si la réponse est `SUCCESS`, vous êtes responsable de la destruction de `retval` (ou alors, retournez la comme valeur de réponse de votre fonction). Si la réponse est `FAILURE`, la valeur de `retval` est indéfinie, et vous ne devez pas y toucher.

Toutes les fonctions internes qui appellent une fonction utilisateur, *DOIVENT* être réentrant. En particulier, elles ne doivent pas utiliser de valeurs globales, ou de variables statiques.

`@xref{function.call-user-function , , call_user_function()}` prend 6 arguments :

C.2.1 HashTable *function_table

[\[Notes en ligne\]](#)

La table de hash dans laquelle la fonction doit être recherchée.

C.2.2 pval *object

[\[Notes en ligne\]](#)

Un pointeur sur un objet sur lequel la fonction est invoquée. Il devrait être à NULL, si on invoque une fonction globale. Si il n'est pas à NULL (ie, il pointe sur un objet), l'argument `function_table` est ignorée, et la liste des fonctions sera lue dans l'objet, plutôt que dans l'argument. L'objet PEUT être modifié par la fonction qui est appelée (la fonction y aura accès via `$this`). Si, pour quelque raison, vous ne le voulez pas, envoyez une copie de l'objet à la place.

C.2.3 pval *function_name

[\[Notes en ligne\]](#)

Le nom de la fonction à appeler. Elle doit être de type `pval`, `IS_STRING`, avec les valeurs de `function_name.str.val` et `function_name.str.len` correctes. `function_name` est modifié par `@xref{function.call-user-function, , call_user_function()}` – il est converti en minuscule. Si vous voulez préserver la casse, envoyez une copie du nom de la fonction.

C.2.4 pval *retval

[\[Notes en ligne\]](#)

Un pointeur sur une structure `pval`, dans laquelle la valeur de retour de la fonction sera placée. La structure doit avoir été allouée au préalable, – `@xref{function.call-user-function, , call_user_function()}` ne l'allouera pas.

C.2.5 int param_count

[\[Notes en ligne\]](#)

Le nombre de paramètre passé à la fonction.

C.2.6 pval *params[]

[\[Notes en ligne\]](#)

Un tableau de pointeur sur les valeurs qui vont être passées comme arguments à la fonction. Le premier argument est à l'offset 0, le second à l'offset 1,... Le tableau est un tableau de pointeurs sur `pval`; Les pointeurs sont envoyés tels quels à la fonction, ce qui signifie que si la fonction modifie les arguments, les valeurs originales seront modifiées. Si vous voulez l'éviter, passez une copie à la place.

C.3 Rapport d'erreurs

[\[Notes en ligne\]](#)

Pour signaler les erreurs d'une fonction interne, vous devez appeler la fonction `@xref{function.php3-error, , php3_error()}`. Cette fonction prend deux arguments au moins : le niveau de l'erreur, et le message d'erreur, sous forme de chaîne de caractères. Tous les arguments suivants sont des paramètres de formats de chaîne. Les niveaux d'erreurs sont :

C.3.1 E_NOTICE

[\[Notes en ligne\]](#)

Notes ne sont pas affichées par défaut, et indique que le script a rencontré quelque chose qui peut être une erreur, mais peut aussi être un événement normal dans la vie du script. Par exemple, essayer d'accéder à une

valeur qui n'a pas été déclarée, ou appeler [stat\(\)](#) sur un fichier qui n'existe pas.

C.3.2 E_WARNING

[\[Notes en ligne\]](#)

Les alertes sont affichées par défaut, mais n'interrompent pas l'exécution du script. Elles indiquent un problème qui doit être intercepté par le script avant que l'appel . Par exemple, appeler [ereg\(\)](#) avec une regex invalide.

C.3.3 E_ERROR

[\[Notes en ligne\]](#)

Les erreurs sont aussi affichées par défaut, et l'exécution du script est interrompue. Elles indiquent des erreurs qui ne peuvent pas être ignorées, comme des problèmes d'allocation de mémoire, par exemple.

C.3.4 E_PARSE

[\[Notes en ligne\]](#)

Les erreurs d'analyse de doivent être générées que par l'analyseur. Elles ne sont citées ici que dans le but d'être exhaustif.

C.3.5 E_CORE_ERROR

[\[Notes en ligne\]](#)

Elles sont similaires aux erreurs E_ERROR, mais elles sont générées par le code de PHP. Les fonctions ne doivent pas générer ce genre d'erreur.

C.3.6 E_CORE_WARNING

[\[Notes en ligne\]](#)

Elles sont similaires à E_WARNING, mais elles sont générées par le code de PHP. Les fonctions ne doivent pas générer ce genre d'erreur.

C.3.7 E_USER_ERROR

[\[Notes en ligne\]](#)

E_USER_ERROR est comparable à E_ERROR. Elle est générée en PHP par l'utilisation de la fonction [trigger_error\(\)](#). Les fonctions ne doivent pas générer ce genre d'erreur.

C.3.8 E_USER_WARNING

[\[Notes en ligne\]](#)

E_USER_WARNING est comparable à E_WARNING. Elle est générée en PHP par l'utilisation de la fonction [trigger_error\(\)](#). Les fonctions ne doivent pas générer ce genre d'erreur.

C.3.9 E_USER_NOTICE

[\[Notes en ligne\]](#)

E_USER_WARNING est comparable à E_NOTICE. Elle est générée en PHP par l'utilisation de la fonction [trigger_error\(\)](#). Les fonctions ne doivent pas générer ce genre d'erreur.

Index des fonctions

[\[Notes en ligne\]](#) [\[Exemples\]](#)

7. [@ref{function.include . , include\(\)}](#)
8. [@ref{function.require . , require\(\)}](#)

A

9. [Abs](#)
10. [accept_connect](#)
11. [Acos](#)
12. [AddCSlashes](#)
13. [AddSlashes](#)
14. [apache_lookup_uri](#)
15. [apache_note](#)
16. [array](#)
17. [array_count_values](#)
18. [array_diff](#)
19. [array_flip](#)
20. [array_intersect](#)
21. [array_keys](#)
22. [array_merge](#)
23. [array_merge_recursive](#)
24. [array_multisort](#)
25. [array_pad](#)
26. [array_pop](#)
27. [array_push](#)
28. [array_rand](#)
29. [array_reverse](#)
30. [array_shift](#)
31. [array_slice](#)
32. [array_splice](#)
33. [array_unique](#)
34. [array_unshift](#)
35. [array_values](#)
36. [array_walk](#)
37. [arsort](#)
38. [ascii2ebcdic](#)
39. [Asin](#)
40. [asort](#)
41. [aspell_check](#)
42. [aspell_check-raw](#)
43. [aspell_new](#)
44. [aspell_suggest](#)
45. [Atan](#)
46. [Atan2](#)

B

47. [base64_decode](#)
48. [base64_encode](#)
49. [base_convert](#)
50. [basename](#)
51. [bcadd](#)
52. [bccomp](#)
53. [bcddiv](#)
54. [bcmmod](#)
55. [bcmul](#)
56. [bcpow](#)
57. [bcscale](#)
58. [bcsqrt](#)
59. [bcsub](#)
60. [bin2hex](#)
61. [bind](#)
62. [BinDec](#)
63. [bindtextdomain](#)

C

64. [call_user_func](#)
65. [Ceil](#)
66. [chdir](#)
67. [checkdate](#)
68. [checkdnsrr](#)
69. [chgrp](#)
70. [chmod](#)
71. [Chop](#)
72. [chown](#)
73. [Chr](#)
74. [chunk_split](#)
75. [clearstatcache](#)
76. [closedir](#)
77. [closelog](#)
78. [com_get](#)
79. [com_invoke](#)
80. [com_load](#)
81. [com_propget](#)
82. [com_propput](#)
83. [com_propset](#)
84. [com_set](#)
85. [compact](#)
86. [connect](#)
87. [connection_aborted](#)
88. [connection_status](#)
89. [connection_timeout](#)
90. [convert_cyr_string](#)
91. [copy](#)

92. [Cos](#)
93. [count](#)
94. [count_chars](#)
95. [cpdf_add_annotation](#)
96. [cpdf_add_outline](#)
97. [cpdf_arc](#)
98. [cpdf_begin_text](#)
99. [cpdf_circle](#)
100. [cpdf_clip](#)
101. [cpdf_close](#)
102. [cpdf_closepath](#)
103. [cpdf_closepath_fill_stroke](#)
104. [cpdf_closepath_stroke](#)
105. [cpdf_continue_text](#)
106. [cpdf_curveto](#)
107. [cpdf_end_text](#)
108. [cpdf_fill](#)
109. [cpdf_fill_stroke](#)
110. [cpdf_finalize](#)
111. [cpdf_finalize_page](#)
112. [cpdf_global_set_document_limits](#)
113. [cpdf_import_jpeg](#)
114. [cpdf_lineto](#)
115. [cpdf_moveto](#)
116. [cpdf_open](#)
117. [cpdf_output_buffer](#)
118. [cpdf_page_init](#)
119. [cpdf_place_inline_image](#)
120. [cpdf_rect](#)
121. [cpdf_restore](#)
122. [cpdf_rlineto](#)
123. [cpdf_rmoveto](#)
124. [cpdf_rotate](#)
125. [cpdf_save](#)
126. [cpdf_save_to_file](#)
127. [cpdf_scale](#)
128. [cpdf_set_char_spacing](#)
129. [cpdf_set_creator](#)
130. [cpdf_set_current_page](#)
131. [cpdf_set_font](#)
132. [cpdf_set_horiz_scaling](#)
133. [cpdf_set_keywords](#)
134. [cpdf_set_leading](#)
135. [cpdf_set_page_animation](#)
136. [cpdf_set_subject](#)
137. [cpdf_set_text_matrix](#)
138. [cpdf_set_text_pos](#)
139. [cpdf_set_text_rendering](#)
140. [cpdf_set_text_rise](#)
141. [cpdf_set_title](#)
142. [cpdf_set_word_spacing](#)

- 143. [cpdf_setdash](#)
- 144. [cpdf_setflat](#)
- 145. [cpdf_setgray](#)
- 146. [cpdf_setgray_fill](#)
- 147. [cpdf_setgray_stroke](#)
- 148. [cpdf_setlinecap](#)
- 149. [cpdf_setlinejoin](#)
- 150. [cpdf_setlinewidth](#)
- 151. [cpdf_setmiterlimit](#)
- 152. [cpdf_setrgbcolor](#)
- 153. [cpdf_setrgbcolor_fill](#)
- 154. [cpdf_setrgbcolor_stroke](#)
- 155. [cpdf_show](#)
- 156. [cpdf_show_xy](#)
- 157. [cpdf_stringwidth](#)
- 158. [cpdf_stroke](#)
- 159. [cpdf_text](#)
- 160. [cpdf_translate](#)
- 161. [create_function](#)
- 162. [crypt](#)
- 163. [curl_close](#)
- 164. [curl_exec](#)
- 165. [curl_init](#)
- 166. [curl_setopt](#)
- 167. [curl_version](#)
- 168. [current](#)
- 169. [cybercash_base64_decode](#)
- 170. [cybercash_base64_encode](#)
- 171. [cybercash_decr](#)
- 172. [cybercash_encr](#)

D

- 173. [date](#)
- 174. [dba_close](#)
- 175. [dba_delete](#)
- 176. [dba_exists](#)
- 177. [dba_fetch](#)
- 178. [dba_firstkey](#)
- 179. [dba_insert](#)
- 180. [dba_nextkey](#)
- 181. [dba_open](#)
- 182. [dba_optimize](#)
- 183. [dba_popen](#)
- 184. [dba_replace](#)
- 185. [dba_sync](#)
- 186. [dbase_add_record](#)
- 187. [dbase_close](#)
- 188. [dbase_create](#)
- 189. [dbase_delete_record](#)

190. [dbase_get_record](#)
191. [dbase_get_record_with_names](#)
192. [dbase_numfields](#)
193. [dbase_numrecords](#)
194. [dbase_open](#)
195. [dbase_pack](#)
196. [dbase_replace_record](#)
197. [dblist](#)
198. [dbmclose](#)
199. [dbmdelete](#)
200. [dbmexists](#)
201. [dbmfetch](#)
202. [dbmfirstkey](#)
203. [dbminsert](#)
204. [dbmnextkey](#)
205. [dbmopen](#)
206. [dbmreplace](#)
207. [dcgettext](#)
208. [debugger_off](#)
209. [debugger_on](#)
210. [DecBin](#)
211. [DecHex](#)
212. [DecOct](#)
213. [define](#)
214. [define_syslog_variables](#)
215. [defined](#)
216. [delete](#)
217. [dgettext](#)
218. [die](#)
219. [dir](#)
220. [dirname](#)
221. [diskfreespace](#)
222. [dl](#)
223. [doubleval](#)

E

224. [each](#)
225. [easter_date](#)
226. [easter_days](#)
227. [ebcdic2ascii](#)
228. [echo](#)
229. [empty](#)
230. [end](#)
231. [ereg](#)
232. [ereg_replace](#)
233. [eregi](#)
234. [eregi_replace](#)
235. [error_log](#)
236. [error_reporting](#)

- 237. [escapeshellcmd](#)
- 238. [eval](#)
- 239. [exec](#)
- 240. [exit](#)
- 241. [Exp](#)
- 242. [explode](#)
- 243. [extension_loaded](#)
- 244. [extract](#)
- 245. [ezmlm_hash](#)

F

- 246. [fclose](#)
- 247. [fdf_close](#)
- 248. [fdf_create](#)
- 249. [fdf_get_file](#)
- 250. [fdf_get_status](#)
- 251. [fdf_get_value](#)
- 252. [fdf_next_field_name](#)
- 253. [fdf_open](#)
- 254. [fdf_save](#)
- 255. [fdf_set_ap](#)
- 256. [fdf_set_file](#)
- 257. [fdf_set_status](#)
- 258. [fdf_set_value](#)
- 259. [feof](#)
- 260. [fgetc](#)
- 261. [fgetcsv](#)
- 262. [fgets](#)
- 263. [fgetss](#)
- 264. [file](#)
- 265. [file_exists](#)
- 266. [fileatime](#)
- 267. [filectime](#)
- 268. [filegroup](#)
- 269. [fileinode](#)
- 270. [filemtime](#)
- 271. [fileowner](#)
- 272. [fileperms](#)
- 273. [filepro](#)
- 274. [filepro_fieldcount](#)
- 275. [filepro_fieldname](#)
- 276. [filepro_fieldtype](#)
- 277. [filepro_fieldwidth](#)
- 278. [filepro_retrieve](#)
- 279. [filepro_rowcount](#)
- 280. [filesize](#)
- 281. [filetype](#)
- 282. [flock](#)
- 283. [Floor](#)

- 284. [flush](#)
- 285. [fopen](#)
- 286. [fpassthru](#)
- 287. [fputs](#)
- 288. [fread](#)
- 289. [FrenchToJD](#)
- 290. [fscanf](#)
- 291. [fseek](#)
- 292. [fsockopen](#)
- 293. [fstat](#)
- 294. [ftell](#)
- 295. [ftp_cdup](#)
- 296. [ftp_chdir](#)
- 297. [ftp_connect](#)
- 298. [ftp_delete](#)
- 299. [ftp_fget](#)
- 300. [ftp_fput](#)
- 301. [ftp_get](#)
- 302. [ftp_login](#)
- 303. [ftp_mdtm](#)
- 304. [ftp_mkdir](#)
- 305. [ftp_nlist](#)
- 306. [ftp_pasv](#)
- 307. [ftp_put](#)
- 308. [ftp_pwd](#)
- 309. [ftp_quit](#)
- 310. [ftp_rawlist](#)
- 311. [ftp_rename](#)
- 312. [ftp_rmdir](#)
- 313. [ftp_site](#)
- 314. [ftp_size](#)
- 315. [ftp_systype](#)
- 316. [ftruncate](#)
- 317. [func_get_arg](#)
- 318. [func_get_args](#)
- 319. [func_num_args](#)
- 320. [function_exists](#)
- 321. [fwrite](#)

G

- 322. [get_browser](#)
- 323. [get_cfg_var](#)
- 324. [get_class_methods](#)
- 325. [get_class_vars](#)
- 326. [get_current_user](#)
- 327. [get_html_translation_table](#)
- 328. [get_magic_quotes_gpc](#)
- 329. [get_magic_quotes_runtime](#)
- 330. [get_meta_tags](#)

- 331. [get_object_vars](#)
- 332. [getallheaders](#)
- 333. [getdate](#)
- 334. [getenv](#)
- 335. [gethostbyaddr](#)
- 336. [gethostbyname](#)
- 337. [gethostbyname1](#)
- 338. [GetImageSize](#)
- 339. [getlastmod](#)
- 340. [getmxrr](#)
- 341. [getmyinode](#)
- 342. [getmypid](#)
- 343. [getmyuid](#)
- 344. [getprotobyname](#)
- 345. [getprotobynumber](#)
- 346. [getrandmax](#)
- 347. [getrusage](#)
- 348. [getservbyname](#)
- 349. [getservbyport](#)
- 350. [gettext](#)
- 351. [gettimeofday](#)
- 352. [gettype](#)
- 353. [gmdate](#)
- 354. [gmmktime](#)
- 355. [gmstrftime](#)
- 356. [GregorianToJD](#)
- 357. [gzclose](#)
- 358. [gzeof](#)
- 359. [gzfile](#)
- 360. [gzgetc](#)
- 361. [gzgets](#)
- 362. [gzgetss](#)
- 363. [gzopen](#)
- 364. [gzpassthru](#)
- 365. [gzputs](#)
- 366. [gzread](#)
- 367. [gzrewind](#)
- 368. [gzseek](#)
- 369. [gztell](#)
- 370. [gzwrite](#)

H

- 371. [header](#)
- 372. [HexDec](#)
- 373. [htmlentities](#)
- 374. [htmlspecialchars](#)
- 375. [hw_Array2Objrec](#)
- 376. [hw_Children](#)
- 377. [hw_ChildrenObj](#)

- 378. [hw Close](#)
- 379. [hw Connect](#)
- 380. [hw Cp](#)
- 381. [hw Deleteobject](#)
- 382. [hw DocByAnchor](#)
- 383. [hw DocByAnchorObj](#)
- 384. [hw DocumentAttributes](#)
- 385. [hw DocumentBodyTag](#)
- 386. [hw DocumentContent](#)
- 387. [hw DocumentSetContent](#)
- 388. [hw DocumentSize](#)
- 389. [hw EditText](#)
- 390. [hw Error](#)
- 391. [hw ErrorMessage](#)
- 392. [hw Free Document](#)
- 393. [hw GetAnchors](#)
- 394. [hw GetAnchorsObj](#)
- 395. [hw GetAndLock](#)
- 396. [hw GetChildColl](#)
- 397. [hw GetChildCollObj](#)
- 398. [hw GetChildDocColl](#)
- 399. [hw GetChildDocCollObj](#)
- 400. [hw GetObject](#)
- 401. [hw GetObjectByQuery](#)
- 402. [hw GetObjectByQueryColl](#)
- 403. [hw GetObjectByQueryCollObj](#)
- 404. [hw GetObjectByQueryObj](#)
- 405. [hw GetParents](#)
- 406. [hw GetParentsObj](#)
- 407. [hw GetRemote](#)
- 408. [hw GetRemoteChildren](#)
- 409. [hw GetSrcByDestObj](#)
- 410. [hw GetText](#)
- 411. [hw Identify](#)
- 412. [hw InCollections](#)
- 413. [hw Info](#)
- 414. [hw InsColl](#)
- 415. [hw InsDoc](#)
- 416. [hw InsertDocument](#)
- 417. [hw InsertObject](#)
- 418. [hw mapid](#)
- 419. [hw Modifyobject](#)
- 420. [hw My](#)
- 421. [hw New Document](#)
- 422. [hw Objrec2Array](#)
- 423. [hw OutputDocument](#)
- 424. [hw pConnect](#)
- 425. [hw PipeDocument](#)
- 426. [hw Root](#)
- 427. [hw Unlock](#)
- 428. [hw Username](#)

429. [hw Who](#)

I

- 430. [ibase_bind](#)
- 431. [ibase_close](#)
- 432. [ibase_connect](#)
- 433. [ibase_execute](#)
- 434. [ibase_fetch_object](#)
- 435. [ibase_fetch_row](#)
- 436. [ibase_free_query](#)
- 437. [ibase_free_result](#)
- 438. [ibase_num_fields](#)
- 439. [ibase_pconnect](#)
- 440. [ibase_prepare](#)
- 441. [ibase_query](#)
- 442. [ibase_timefmt](#)
- 443. [icap_close](#)
- 444. [icap_delete_event](#)
- 445. [icap_fetch_event](#)
- 446. [icap_list_alarms](#)
- 447. [icap_list_events](#)
- 448. [icap_open](#)
- 449. [icap_snooze](#)
- 450. [icap_store_event](#)
- 451. [ifx_affected_rows](#)
- 452. [ifx_blobinfile_mode](#)
- 453. [ifx_byteasvarchar](#)
- 454. [ifx_close](#)
- 455. [ifx_connect](#)
- 456. [ifx_copy_blob](#)
- 457. [ifx_create_blob](#)
- 458. [ifx_create_char](#)
- 459. [ifx_do](#)
- 460. [ifx_error](#)
- 461. [ifx_errormsg](#)
- 462. [ifx_fetch_row](#)
- 463. [ifx_fieldproperties](#)
- 464. [ifx_fieldtypes](#)
- 465. [ifx_free_blob](#)
- 466. [ifx_free_char](#)
- 467. [ifx_free_result](#)
- 468. [ifx_free_slob](#)
- 469. [ifx_get_blob](#)
- 470. [ifx_get_char](#)
- 471. [ifx_getsqlca](#)
- 472. [ifx_htmltbl_result](#)
- 473. [ifx_nullformat](#)
- 474. [ifx_num_fields](#)
- 475. [ifx_num_rows](#)

I

- 476. [ifx_pconnect](#)
- 477. [ifx_prepare](#)
- 478. [ifx_query](#)
- 479. [ifx_textasvarchar](#)
- 480. [ifx_update_blob](#)
- 481. [ifx_update_char](#)
- 482. [ifxus_close_slob](#)
- 483. [ifxus_create_slob](#)
- 484. [ifxus_open_slob](#)
- 485. [ifxus_read_slob](#)
- 486. [ifxus_seek_slob](#)
- 487. [ifxus_tell_slob](#)
- 488. [ifxus_write_slob](#)
- 489. [ignore_user_abort](#)
- 490. [ImageArc](#)
- 491. [ImageChar](#)
- 492. [ImageCharUp](#)
- 493. [ImageColorAllocate](#)
- 494. [ImageColorAt](#)
- 495. [ImageColorClosest](#)
- 496. [ImageColorDeAllocate](#)
- 497. [ImageColorExact](#)
- 498. [ImageColorResolve](#)
- 499. [ImageColorSet](#)
- 500. [ImageColorsForIndex](#)
- 501. [ImageColorsTotal](#)
- 502. [ImageColorTransparent](#)
- 503. [ImageCopy](#)
- 504. [ImageCopyResized](#)
- 505. [ImageCreate](#)
- 506. [ImageCreateFromGif](#)
- 507. [ImageCreateFromJPEG](#)
- 508. [ImageCreateFromPNG](#)
- 509. [ImageDashedLine](#)
- 510. [ImageDestroy](#)
- 511. [ImageFill](#)
- 512. [ImageFilledPolygon](#)
- 513. [ImageFilledRectangle](#)
- 514. [ImageFillToBorder](#)
- 515. [ImageFontHeight](#)
- 516. [ImageFontWidth](#)
- 517. [ImageGammaCorrect](#)
- 518. [ImageGif](#)
- 519. [ImageInterlace](#)
- 520. [ImageJPEG](#)
- 521. [ImageLine](#)
- 522. [ImageLoadFont](#)
- 523. [ImagePNG](#)
- 524. [ImagePolygon](#)
- 525. [ImagePSBBox](#)
- 526. [ImagePSEncodeFont](#)

- 527. [ImagePsExtendFont](#)
- 528. [ImagePSFreeFont](#)
- 529. [ImagePSLoadFont](#)
- 530. [ImagePsSlantFont](#)
- 531. [ImagePSText](#)
- 532. [ImageRectangle](#)
- 533. [ImageSetPixel](#)
- 534. [ImageString](#)
- 535. [ImageStringUp](#)
- 536. [ImageSX](#)
- 537. [ImageSY](#)
- 538. [ImageTTFBBox](#)
- 539. [ImageTTFTText](#)
- 540. [ImageTypes](#)
- 541. [imap_8bit](#)
- 542. [imap_alerts](#)
- 543. [imap_append](#)
- 544. [imap_base64](#)
- 545. [imap_binary](#)
- 546. [imap_body](#)
- 547. [imap_check](#)
- 548. [imap_clearflag_full](#)
- 549. [imap_close](#)
- 550. [imap_createmailbox](#)
- 551. [imap_delete](#)
- 552. [imap_deletemailbox](#)
- 553. [imap_errors](#)
- 554. [imap_expunge](#)
- 555. [imap_fetchbody](#)
- 556. [imap_fetchheader](#)
- 557. [imap_fetchstructure](#)
- 558. [imap_getmailboxes](#)
- 559. [imap_getsubscribed](#)
- 560. [imap_header](#)
- 561. [imap_headers](#)
- 562. [imap_last_error](#)
- 563. [imap_listmailbox](#)
- 564. [imap_listsubscribed](#)
- 565. [imap_mail_copy](#)
- 566. [imap_mail_move](#)
- 567. [imap_mailboxmsginfo](#)
- 568. [imap_msgno](#)
- 569. [imap_num_msg](#)
- 570. [imap_num_recent](#)
- 571. [imap_open](#)
- 572. [imap_ping](#)
- 573. [imap_qprint](#)
- 574. [imap_renamemailbox](#)
- 575. [imap_reopen](#)
- 576. [imap_rfc822_parse_adrlist](#)
- 577. [imap_rfc822_write_address](#)

- 578. [imap_scanmailbox](#)
- 579. [imap_search](#)
- 580. [imap_setflag_full](#)
- 581. [imap_sort](#)
- 582. [imap_status](#)
- 583. [imap_subscribe](#)
- 584. [imap_uid](#)
- 585. [imap_undelete](#)
- 586. [imap_unsubscribe](#)
- 587. [imap_utf7_decode](#)
- 588. [imap_utf7_encode](#)
- 589. [implode](#)
- 590. [in_array](#)
- 591. [intval](#)
- 592. [ip2long](#)
- 593. [iptcparse](#)
- 594. [is_array](#)
- 595. [is_dir](#)
- 596. [is_double](#)
- 597. [is_executable](#)
- 598. [is_file](#)
- 599. [is_float](#)
- 600. [is_int](#)
- 601. [is_integer](#)
- 602. [is_link](#)
- 603. [is_long](#)
- 604. [is_object](#)
- 605. [is_readable](#)
- 606. [is_real](#)
- 607. [is_string](#)
- 608. [is_uploaded_file](#)
- 609. [is_writable](#)
- 610. [isset](#)

J

- 611. [JDDayOfWeek](#)
- 612. [JDMonthName](#)
- 613. [JDToFrench](#)
- 614. [JDToGregorian](#)
- 615. [JDToJewish](#)
- 616. [JDToJulian](#)
- 617. [JewishToJD](#)
- 618. [join](#)
- 619. [JulianToJD](#)

K

- 620. [key](#)
- 621. [krsort](#)

622. [ksort](#)

L

623. [ldap_add](#)
624. [ldap_bind](#)
625. [ldap_close](#)
626. [ldap_connect](#)
627. [ldap_count_entries](#)
628. [ldap_delete](#)
629. [ldap_dn2ufn](#)
630. [ldap_err2str](#)
631. [ldap_errno](#)
632. [ldap_error](#)
633. [ldap_explode_dn](#)
634. [ldap_first_attribute](#)
635. [ldap_first_entry](#)
636. [ldap_free_result](#)
637. [ldap_get_attributes](#)
638. [ldap_get_dn](#)
639. [ldap_get_entries](#)
640. [ldap_get_values](#)
641. [ldap_get_values_len](#)
642. [ldap_list](#)
643. [ldap_mod_add](#)
644. [ldap_mod_del](#)
645. [ldap_mod_replace](#)
646. [ldap_modify](#)
647. [ldap_next_attribute](#)
648. [ldap_next_entry](#)
649. [ldap_read](#)
650. [ldap_search](#)
651. [ldap_unbind](#)
652. [leak](#)
653. [link](#)
654. [linkinfo](#)
655. [list](#)
656. [listen](#)
657. [Log](#)
658. [Log10](#)
659. [long2ip](#)
660. [lstat](#)
661. [ltrim](#)

M

662. [mail](#)
663. [max](#)
664. [mcal_append_event](#)
665. [mcal_close](#)

- 666. [mcal_date_compare](#)
- 667. [mcal_date_valid](#)
- 668. [mcal_day_of_week](#)
- 669. [mcal_day_of_year](#)
- 670. [mcal_days_in_month](#)
- 671. [mcal_delete_event](#)
- 672. [mcal_event_init](#)
- 673. [mcal_event_set_alarm](#)
- 674. [mcal_event_set_category](#)
- 675. [mcal_event_set_class](#)
- 676. [mcal_event_set_description](#)
- 677. [mcal_event_set_end](#)
- 678. [mcal_event_set_recur_daily](#)
- 679. [mcal_event_set_recur_monthly_mday](#)
- 680. [mcal_event_set_recur_monthly_wday](#)
- 681. [mcal_event_set_recur_none](#)
- 682. [mcal_event_set_recur_weekly](#)
- 683. [mcal_event_set_recur_yearly](#)
- 684. [mcal_event_set_start](#)
- 685. [mcal_event_set_title](#)
- 686. [mcal_fetch_current_stream_event](#)
- 687. [mcal_fetch_event](#)
- 688. [mcal_is_leap_year](#)
- 689. [mcal_list_alarms](#)
- 690. [mcal_list_events](#)
- 691. [mcal_next_recurrence](#)
- 692. [mcal_open](#)
- 693. [mcal_snooze](#)
- 694. [mcal_store_event](#)
- 695. [mcal_time_valid](#)
- 696. [mdecrypt_cbc](#)
- 697. [mdecrypt_cfb](#)
- 698. [mdecrypt_create_iv](#)
- 699. [mdecrypt_decrypt](#)
- 700. [mdecrypt_ecb](#)
- 701. [mdecrypt_enc_get_algorithms_name](#)
- 702. [mdecrypt_enc_get_block_size](#)
- 703. [mdecrypt_enc_get_iv_size](#)
- 704. [mdecrypt_enc_get_key_size](#)
- 705. [mdecrypt_enc_get_modes_name](#)
- 706. [mdecrypt_enc_get_supported_key_sizes](#)
- 707. [mdecrypt_enc_is_block_algorithm](#)
- 708. [mdecrypt_enc_is_block_algorithm_mode](#)
- 709. [mdecrypt_enc_is_block_mode](#)
- 710. [mdecrypt_enc_self_test](#)
- 711. [mdecrypt_encrypt](#)
- 712. [mdecrypt_generic](#)
- 713. [mdecrypt_generic_end](#)
- 714. [mdecrypt_generic_init](#)
- 715. [mdecrypt_get_block_size](#)
- 716. [mdecrypt_get_cipher_name](#)

- 717. [mdecrypt_get_iv_size](#)
- 718. [mdecrypt_get_key_size](#)
- 719. [mdecrypt_list_algorithms](#)
- 720. [mdecrypt_list_modes](#)
- 721. [mdecrypt_module_get_algo_block_size](#)
- 722. [mdecrypt_module_get_algo_key_size](#)
- 723. [mdecrypt_module_get_algo_supported_key_sizes](#)
- 724. [mdecrypt_module_is_block_algorithm](#)
- 725. [mdecrypt_module_is_block_algorithm_mode](#)
- 726. [mdecrypt_module_is_block_mode](#)
- 727. [mdecrypt_module_open](#)
- 728. [mdecrypt_module_self_test](#)
- 729. [mdecrypt_ofb](#)
- 730. [md5](#)
- 731. [mdecrypt_generic](#)
- 732. [Metaphone](#)
- 733. [method_exists](#)
- 734. [mhash](#)
- 735. [mhash_count](#)
- 736. [mhash_get_block_size](#)
- 737. [mhash_get_hash_name](#)
- 738. [mhash_keygen_s2k](#)
- 739. [microtime](#)
- 740. [min](#)
- 741. [mkdir](#)
- 742. [mktime](#)
- 743. [move_uploaded_file](#)
- 744. [mysql](#)
- 745. [mysql_affected_rows](#)
- 746. [mysql_close](#)
- 747. [mysql_connect](#)
- 748. [mysql_create_db](#)
- 749. [mysql_createdb](#)
- 750. [mysql_data_seek](#)
- 751. [mysql_dbname](#)
- 752. [mysql_drop_db](#)
- 753. [mysql_dropdb](#)
- 754. [mysql_error](#)
- 755. [mysql_fetch_array](#)
- 756. [mysql_fetch_field](#)
- 757. [mysql_fetch_object](#)
- 758. [mysql_fetch_row](#)
- 759. [mysql_field_seek](#)
- 760. [mysql_fieldflags](#)
- 761. [mysql_fieldlen](#)
- 762. [mysql_fieldname](#)
- 763. [mysql_fieldtable](#)
- 764. [mysql_fieldtype](#)
- 765. [mysql_free_result](#)
- 766. [mysql_freeresult](#)
- 767. [mysql_list_dbs](#)

- 768. [mysql_list_fields](#)
- 769. [mysql_list_tables](#)
- 770. [mysql_listdbs](#)
- 771. [mysql_listfields](#)
- 772. [mysql_listtables](#)
- 773. [mysql_num_fields](#)
- 774. [mysql_num_rows](#)
- 775. [mysql_numfields](#)
- 776. [mysql_numrows](#)
- 777. [mysql_pconnect](#)
- 778. [mysql_query](#)
- 779. [mysql_regcase](#)
- 780. [mysql_result](#)
- 781. [mysql_select_db](#)
- 782. [mysql_selectdb](#)
- 783. [mysql_tablename](#)
- 784. [mssql_close](#)
- 785. [mssql_connect](#)
- 786. [mssql_data_seek](#)
- 787. [mssql_fetch_array](#)
- 788. [mssql_fetch_field](#)
- 789. [mssql_fetch_object](#)
- 790. [mssql_fetch_row](#)
- 791. [mssql_field_length](#)
- 792. [mssql_field_name](#)
- 793. [mssql_field_seek](#)
- 794. [mssql_field_type](#)
- 795. [mssql_free_result](#)
- 796. [mssql_get_last_message](#)
- 797. [mssql_min_error_severity](#)
- 798. [mssql_min_message_severity](#)
- 799. [mssql_num_fields](#)
- 800. [mssql_num_rows](#)
- 801. [mssql_pconnect](#)
- 802. [mssql_query](#)
- 803. [mssql_result](#)
- 804. [mssql_select_db](#)
- 805. [mt_getrandmax](#)
- 806. [mt_rand](#)
- 807. [mt_srand](#)
- 808. [mysql_affected_rows](#)
- 809. [mysql_change_user](#)
- 810. [mysql_close](#)
- 811. [mysql_connect](#)
- 812. [mysql_create_db](#)
- 813. [mysql_data_seek](#)
- 814. [mysql_db_query](#)
- 815. [mysql_drop_db](#)
- 816. [mysql_errno](#)
- 817. [mysql_error](#)
- 818. [mysql_fetch_array](#)

- 819. [mysql_fetch_field](#)
- 820. [mysql_fetch_lengths](#)
- 821. [mysql_fetch_object](#)
- 822. [mysql_fetch_row](#)
- 823. [mysql_field_flags](#)
- 824. [mysql_field_len](#)
- 825. [mysql_field_name](#)
- 826. [mysql_field_seek](#)
- 827. [mysql_field_table](#)
- 828. [mysql_field_type](#)
- 829. [mysql_free_result](#)
- 830. [mysql_insert_id](#)
- 831. [mysql_list_dbs](#)
- 832. [mysql_list_fields](#)
- 833. [mysql_list_tables](#)
- 834. [mysql_num_fields](#)
- 835. [mysql_num_rows](#)
- 836. [mysql_pconnect](#)
- 837. [mysql_query](#)
- 838. [mysql_result](#)
- 839. [mysql_select_db](#)
- 840. [mysql_tablename](#)

N

- 841. [natcasesort](#)
- 842. [natsort](#)
- 843. [next](#)
- 844. [nl2br](#)
- 845. [number_format](#)

O

- 846. [ob_end_clean](#)
- 847. [ob_end_flush](#)
- 848. [ob_get_contents](#)
- 849. [ob_get_length](#)
- 850. [ob_implicit_flush](#)
- 851. [ob_start](#)
- 852. [OCIBindByName](#)
- 853. [OCIColumnIsNULL](#)
- 854. [OCIColumnName](#)
- 855. [OCIColumnSize](#)
- 856. [OCIColumnType](#)
- 857. [OCICommit](#)
- 858. [OCIDefineByName](#)
- 859. [OCIError](#)
- 860. [OCIExecute](#)
- 861. [OCIFetch](#)
- 862. [OCIFetchInto](#)

- 863. [OCIFetchStatement](#)
- 864. [OCIFreeCursor](#)
- 865. [OCIFreeStatement](#)
- 866. [OCIInternalDebug](#)
- 867. [OCILogOff](#)
- 868. [OCILogon](#)
- 869. [OCINewCursor](#)
- 870. [OCINewDescriptor](#)
- 871. [OCINLogon](#)
- 872. [OCINumCols](#)
- 873. [OCIParse](#)
- 874. [OCIPLogon](#)
- 875. [OCIResult](#)
- 876. [OCIRollback](#)
- 877. [OCIRowCount](#)
- 878. [OCIServerVersion](#)
- 879. [OCIStatementType](#)
- 880. [OctDec](#)
- 881. [odbc_autocommit](#)
- 882. [odbc_binmode](#)
- 883. [odbc_close](#)
- 884. [odbc_close_all](#)
- 885. [odbc_commit](#)
- 886. [odbc_connect](#)
- 887. [odbc_cursor](#)
- 888. [odbc_do](#)
- 889. [odbc_exec](#)
- 890. [odbc_execute](#)
- 891. [odbc_fetch_into](#)
- 892. [odbc_fetch_row](#)
- 893. [odbc_field_len](#)
- 894. [odbc_field_name](#)
- 895. [odbc_field_type](#)
- 896. [odbc_free_result](#)
- 897. [odbc_longreadlen](#)
- 898. [odbc_num_fields](#)
- 899. [odbc_num_rows](#)
- 900. [odbc_pconnect](#)
- 901. [odbc_prepare](#)
- 902. [odbc_result](#)
- 903. [odbc_result_all](#)
- 904. [odbc_rollback](#)
- 905. [odbc_setoption](#)
- 906. [opendir](#)
- 907. [openlog](#)
- 908. [Ora_Bind](#)
- 909. [Ora_Close](#)
- 910. [Ora_ColumnName](#)
- 911. [Ora_ColumnType](#)
- 912. [Ora_Commit](#)
- 913. [Ora_CommitOff](#)

- 914. [Ora CommitOn](#)
- 915. [Ora Error](#)
- 916. [Ora ErrorCode](#)
- 917. [Ora Exec](#)
- 918. [Ora Fetch](#)
- 919. [Ora GetColumn](#)
- 920. [Ora Logoff](#)
- 921. [Ora Logon](#)
- 922. [Ora Open](#)
- 923. [Ora Parse](#)
- 924. [Ora Rollback](#)
- 925. [Ord](#)
- 926. [ovrimos_close](#)
- 927. [ovrimos_close_all](#)
- 928. [ovrimos_commit](#)
- 929. [ovrimos_connect](#)
- 930. [ovrimos_cursor](#)
- 931. [ovrimos_exec](#)
- 932. [ovrimos_execute](#)
- 933. [ovrimos_fetch_into](#)
- 934. [ovrimos_fetch_row](#)
- 935. [ovrimos_field_len](#)
- 936. [ovrimos_field_name](#)
- 937. [ovrimos_field_num](#)
- 938. [ovrimos_field_type](#)
- 939. [ovrimos_free_result](#)
- 940. [ovrimos_longreadlen](#)
- 941. [ovrimos_num_fields](#)
- 942. [ovrimos_num_rows](#)
- 943. [ovrimos_prepare](#)
- 944. [ovrimos_result](#)
- 945. [ovrimos_result_all](#)
- 946. [ovrimos_rollback](#)

P

- 947. [pack](#)
- 948. [parse_str](#)
- 949. [parse_url](#)
- 950. [passthru](#)
- 951. [pclose](#)
- 952. [pdf_add_annotation](#)
- 953. [PDF_add_outline](#)
- 954. [PDF_arc](#)
- 955. [PDF_begin_page](#)
- 956. [PDF_circle](#)
- 957. [PDF_clip](#)
- 958. [PDF_close](#)
- 959. [PDF_close_image](#)
- 960. [PDF_closepath](#)

- 961. [PDF closepath fill stroke](#)
- 962. [PDF closepath stroke](#)
- 963. [PDF continue text](#)
- 964. [PDF curveto](#)
- 965. [PDF end page](#)
- 966. [PDF endpath](#)
- 967. [PDF execute image](#)
- 968. [PDF fill](#)
- 969. [PDF fill stroke](#)
- 970. [PDF get info](#)
- 971. [PDF lineto](#)
- 972. [PDF moveto](#)
- 973. [PDF open](#)
- 974. [PDF open gif](#)
- 975. [PDF open jpeg](#)
- 976. [PDF open memory image](#)
- 977. [PDF place image](#)
- 978. [PDF put image](#)
- 979. [PDF rect](#)
- 980. [PDF restore](#)
- 981. [PDF rotate](#)
- 982. [PDF save](#)
- 983. [PDF scale](#)
- 984. [PDF set char spacing](#)
- 985. [PDF set duration](#)
- 986. [PDF set font](#)
- 987. [PDF set horiz scaling](#)
- 988. [PDF set info author](#)
- 989. [PDF set info creator](#)
- 990. [PDF set info keywords](#)
- 991. [PDF set info subject](#)
- 992. [PDF set info title](#)
- 993. [PDF set leading](#)
- 994. [PDF set parameter](#)
- 995. [PDF set text matrix](#)
- 996. [PDF set text pos](#)
- 997. [PDF set text rendering](#)
- 998. [PDF set text rise](#)
- 999. [PDF set transition](#)
- 1000. [PDF set word spacing](#)
- 1001. [PDF setdash](#)
- 1002. [PDF setflat](#)
- 1003. [PDF setgray](#)
- 1004. [PDF setgray fill](#)
- 1005. [PDF setgray stroke](#)
- 1006. [PDF setlinecap](#)
- 1007. [PDF setlinejoin](#)
- 1008. [PDF setlinewidth](#)
- 1009. [PDF setmiterlimit](#)
- 1010. [PDF setrgbcolor](#)
- 1011. [PDF setrgbcolor fill](#)

- 1012. [PDF setrgbcolor stroke](#)
- 1013. [PDF show](#)
- 1014. [PDF show boxed](#)
- 1015. [PDF show xy](#)
- 1016. [PDF skew](#)
- 1017. [PDF stringwidth](#)
- 1018. [PDF stroke](#)
- 1019. [PDF translate](#)
- 1020. [pfpro cleanup](#)
- 1021. [pfpro init](#)
- 1022. [pfpro process](#)
- 1023. [pfpro process raw](#)
- 1024. [pfpro version](#)
- 1025. [pfssockopen](#)
- 1026. [pg_Close](#)
- 1027. [pg_cmdTuples](#)
- 1028. [pg_Connect](#)
- 1029. [pg_DBname](#)
- 1030. [pg_ErrorMessage](#)
- 1031. [pg_Exec](#)
- 1032. [pg_Fetch_Array](#)
- 1033. [pg_Fetch_Object](#)
- 1034. [pg_Fetch_Row](#)
- 1035. [pg_FieldIsNull](#)
- 1036. [pg_FieldName](#)
- 1037. [pg_FieldNum](#)
- 1038. [pg_FieldPrtLen](#)
- 1039. [pg_FieldSize](#)
- 1040. [pg_FieldType](#)
- 1041. [pg_FreeResult](#)
- 1042. [pg_GetLastOid](#)
- 1043. [pg_Host](#)
- 1044. [pg_loclose](#)
- 1045. [pg_locreate](#)
- 1046. [pg_loopen](#)
- 1047. [pg_loread](#)
- 1048. [pg_loreadall](#)
- 1049. [pg_lounlink](#)
- 1050. [pg_lowrite](#)
- 1051. [pg_NumFields](#)
- 1052. [pg_NumRows](#)
- 1053. [pg_Options](#)
- 1054. [pg_pConnect](#)
- 1055. [pg_Port](#)
- 1056. [pg_Result](#)
- 1057. [pg_tty](#)
- 1058. [php_logo_guid](#)
- 1059. [phpinfo](#)
- 1060. [phpversion](#)
- 1061. [pi](#)
- 1062. [popen](#)

- 1063. [pos](#)
- 1064. [posix_ctermid](#)
- 1065. [posix_getcwd](#)
- 1066. [posix_getegid](#)
- 1067. [posix_geteuid](#)
- 1068. [posix_getgid](#)
- 1069. [posix_getgrgid](#)
- 1070. [posix_getgrnam](#)
- 1071. [posix_getgroups](#)
- 1072. [posix_getlogin](#)
- 1073. [posix_getpgid](#)
- 1074. [posix_getpgrp](#)
- 1075. [posix_getpid](#)
- 1076. [posix_getppid](#)
- 1077. [posix_getpwnam](#)
- 1078. [posix_getpwuid](#)
- 1079. [posix_getrlimit](#)
- 1080. [posix_getsid](#)
- 1081. [posix_getuid](#)
- 1082. [posix_isatty](#)
- 1083. [posix_kill](#)
- 1084. [posix_mkfifo](#)
- 1085. [posix_setgid](#)
- 1086. [posix_setpgid](#)
- 1087. [posix_setsid](#)
- 1088. [posix_setuid](#)
- 1089. [posix_times](#)
- 1090. [posix_ttyname](#)
- 1091. [posix_uname](#)
- 1092. [pow](#)
- 1093. [preg_grep](#)
- 1094. [preg_match](#)
- 1095. [preg_match_all](#)
- 1096. [preg_quote](#)
- 1097. [preg_replace](#)
- 1098. [preg_split](#)
- 1099. [prev](#)
- 1100. [print](#)
- 1101. [print_r](#)
- 1102. [printf](#)
- 1103. [pspell_check](#)
- 1104. [pspell_mode](#)
- 1105. [pspell_new](#)
- 1106. [pspell_runtogether](#)
- 1107. [pspell_suggest](#)
- 1108. [putenv](#)

Q

- 1109. [quoted_printable_decode](#)

1110. [QuoteMeta](#)

R

1111. [rand](#)
1112. [range](#)
1113. [rawurldecode](#)
1114. [rawurlencode](#)
1115. [read_exif_data](#)
1116. [readdir](#)
1117. [readfile](#)
1118. [readgzfile](#)
1119. [readline](#)
1120. [readline_add_history](#)
1121. [readline_clear_history](#)
1122. [readline_completion_function](#)
1123. [readline_info](#)
1124. [readline_list_history](#)
1125. [readline_read_history](#)
1126. [readline_write_history](#)
1127. [readlink](#)
1128. [realpath](#)
1129. [recode_file](#)
1130. [recode_string](#)
1131. [register_shutdown_function](#)
1132. [rename](#)
1133. [reset](#)
1134. [restore_error_handler](#)
1135. [rewind](#)
1136. [rewinddir](#)
1137. [rmdir](#)
1138. [round](#)
1139. [rsort](#)

S

1140. [satellite_caught_exception](#)
1141. [satellite_exception_id](#)
1142. [satellite_exception_value](#)
1143. [sem_acquire](#)
1144. [sem_get](#)
1145. [sem_release](#)
1146. [serialize](#)
1147. [session_decode](#)
1148. [session_destroy](#)
1149. [session_encode](#)
1150. [session_id](#)
1151. [session_is_registered](#)
1152. [session_module_name](#)
1153. [session_name](#)

- 1154. [session_register](#)
- 1155. [session_save_path](#)
- 1156. [session_start](#)
- 1157. [session_unregister](#)
- 1158. [set_error_handler](#)
- 1159. [set_file_buffer](#)
- 1160. [set_magic_quotes_runtime](#)
- 1161. [set_socket_blocking](#)
- 1162. [set_time_limit](#)
- 1163. [setcookie](#)
- 1164. [setlocale](#)
- 1165. [settype](#)
- 1166. [shm_attach](#)
- 1167. [shm_detach](#)
- 1168. [shm_get_var](#)
- 1169. [shm_put_var](#)
- 1170. [shm_remove](#)
- 1171. [shm_remove_var](#)
- 1172. [shmop_close](#)
- 1173. [shmop_delete](#)
- 1174. [shmop_open](#)
- 1175. [shmop_read](#)
- 1176. [shmop_size](#)
- 1177. [shmop_write](#)
- 1178. [shuffle](#)
- 1179. [similar_text](#)
- 1180. [Sin](#)
- 1181. [sizeof](#)
- 1182. [sleep](#)
- 1183. [snmp_get_quick_print](#)
- 1184. [snmp_set_quick_print](#)
- 1185. [snmpget](#)
- 1186. [snmpset](#)
- 1187. [snmpwalk](#)
- 1188. [snmpwalkoid](#)
- 1189. [socket](#)
- 1190. [socket_get_status](#)
- 1191. [socket_set_timeout](#)
- 1192. [sort](#)
- 1193. [soundex](#)
- 1194. [split](#)
- 1195. [sprintf](#)
- 1196. [sql_regcase](#)
- 1197. [Sqrt](#)
- 1198. [srand](#)
- 1199. [stat](#)
- 1200. [str_repeat](#)
- 1201. [str_replace](#)
- 1202. [strcasecmp](#)
- 1203. [strchr](#)
- 1204. [strcmp](#)

- 1205. [strcspn](#)
- 1206. [strerror](#)
- 1207. [strftime](#)
- 1208. [strip_tags](#)
- 1209. [StripCslashes](#)
- 1210. [StripSlashes](#)
- 1211. [stristr](#)
- 1212. [strlen](#)
- 1213. [strpos](#)
- 1214. [strchr](#)
- 1215. [strrev](#)
- 1216. [strrpos](#)
- 1217. [strspn](#)
- 1218. [strstr](#)
- 1219. [strtok](#)
- 1220. [strtolower](#)
- 1221. [strtoupper](#)
- 1222. [strtr](#)
- 1223. [strval](#)
- 1224. [substr](#)
- 1225. [substr_replace](#)
- 1226. [swf_actiongeturl](#)
- 1227. [swf_actiongotoframe](#)
- 1228. [swf_actiongotolabel](#)
- 1229. [swf_actionnextframe](#)
- 1230. [swf_actionplay](#)
- 1231. [swf_actionprevframe](#)
- 1232. [swf_actionsettarget](#)
- 1233. [swf_actionstop](#)
- 1234. [swf_actiontogglequality](#)
- 1235. [swf_actionwaitforframe](#)
- 1236. [swf_addbuttonrecord](#)
- 1237. [swf_addcolor](#)
- 1238. [swf_closefile](#)
- 1239. [swf_definebitmap](#)
- 1240. [swf_definefont](#)
- 1241. [swf_defineline](#)
- 1242. [swf_definepoly](#)
- 1243. [swf_definerect](#)
- 1244. [swf_definetext](#)
- 1245. [swf_endbutton](#)
- 1246. [swf_enddoaction](#)
- 1247. [swf_endshape](#)
- 1248. [swf_endsymbol](#)
- 1249. [swf_fontsize](#)
- 1250. [swf_fontslant](#)
- 1251. [swf_fontracking](#)
- 1252. [swf_getbitmapinfo](#)
- 1253. [swf_getfontinfo](#)
- 1254. [swf_getframe](#)
- 1255. [swf_labelframe](#)

- 1256. [swf_lookat](#)
- 1257. [swf_modifyobject](#)
- 1258. [swf_mulcolor](#)
- 1259. [swf_nextid](#)
- 1260. [swf_oncondition](#)
- 1261. [swf_openfile](#)
- 1262. [swf_ortho](#)
- 1263. [swf_ortho2](#)
- 1264. [swf_perspective](#)
- 1265. [swf_placeobject](#)
- 1266. [swf_polarview](#)
- 1267. [swf_popmatrix](#)
- 1268. [swf_posround](#)
- 1269. [swf_pushmatrix](#)
- 1270. [swf_removeobject](#)
- 1271. [swf_rotate](#)
- 1272. [swf_scale](#)
- 1273. [swf_setfont](#)
- 1274. [swf_setframe](#)
- 1275. [swf_shapearc](#)
- 1276. [swf_shapecurveto](#)
- 1277. [swf_shapecurveto3](#)
- 1278. [swf_shapefillbitmapclip](#)
- 1279. [swf_shapefillbitmaptile](#)
- 1280. [swf_shapefilloff](#)
- 1281. [swf_shapefillsolid](#)
- 1282. [swf_shapelinesolid](#)
- 1283. [swf_shapelineto](#)
- 1284. [swf_shapemoveto](#)
- 1285. [swf_showframe](#)
- 1286. [swf_startbutton](#)
- 1287. [swf_startdoaction](#)
- 1288. [swf_startshape](#)
- 1289. [swf_startsymbol](#)
- 1290. [swf_textwidth](#)
- 1291. [swf_translate](#)
- 1292. [swf_viewport](#)
- 1293. [sybase_affected_rows](#)
- 1294. [sybase_close](#)
- 1295. [sybase_connect](#)
- 1296. [sybase_data_seek](#)
- 1297. [sybase_fetch_array](#)
- 1298. [sybase_fetch_field](#)
- 1299. [sybase_fetch_object](#)
- 1300. [sybase_fetch_row](#)
- 1301. [sybase_field_seek](#)
- 1302. [sybase_free_result](#)
- 1303. [sybase_get_last_message](#)
- 1304. [sybase_min_client_severity](#)
- 1305. [sybase_min_error_severity](#)
- 1306. [sybase_min_message_severity](#)

- 1307. [sybase_min_server_severity](#)
- 1308. [sybase_num_fields](#)
- 1309. [sybase_num_rows](#)
- 1310. [sybase_pconnect](#)
- 1311. [sybase_query](#)
- 1312. [sybase_result](#)
- 1313. [sybase_select_db](#)
- 1314. [symlink](#)
- 1315. [syslog](#)
- 1316. [system](#)

T

- 1317. [Tan](#)
- 1318. [tempnam](#)
- 1319. [textdomain](#)
- 1320. [time](#)
- 1321. [tmpfile](#)
- 1322. [touch](#)
- 1323. [trigger_error](#)
- 1324. [trim](#)

U

- 1325. [uasort](#)
- 1326. [ucfirst](#)
- 1327. [ucwords](#)
- 1328. [uksort](#)
- 1329. [umask](#)
- 1330. [uniqid](#)
- 1331. [unlink](#)
- 1332. [unpack](#)
- 1333. [unserialize](#)
- 1334. [unset](#)
- 1335. [urldecode](#)
- 1336. [urlencode](#)
- 1337. [user_error](#)
- 1338. [usleep](#)
- 1339. [usort](#)
- 1340. [utf8_decode](#)
- 1341. [utf8_encode](#)

V

- 1342. [var_dump](#)
- 1343. [virtual](#)
- 1344. [vm_addalias](#)
- 1345. [vm_adduser](#)
- 1346. [vm_delalias](#)

- 1347. [vm_deluser](#)
- 1348. [vm_passwd](#)

W

- 1349. [wddx_add_vars](#)
- 1350. [wddx_deserialize](#)
- 1351. [wddx_packet_end](#)
- 1352. [wddx_packet_start](#)
- 1353. [wddx_serialize_value](#)
- 1354. [wddx_serialize_vars](#)

X

- 1355. [xml_error_string](#)
- 1356. [xml_get_current_byte_index](#)
- 1357. [xml_get_current_column_number](#)
- 1358. [xml_get_current_line_number](#)
- 1359. [xml_get_error_code](#)
- 1360. [xml_parse](#)
- 1361. [xml_parser_create](#)
- 1362. [xml_parser_free](#)
- 1363. [xml_parser_get_option](#)
- 1364. [xml_parser_set_option](#)
- 1365. [xml_set_character_data_handler](#)
- 1366. [xml_set_default_handler](#)
- 1367. [xml_set_element_handler](#)
- 1368. [xml_set_external_entity_ref_handler](#)
- 1369. [xml_set_notation_decl_handler](#)
- 1370. [xml_set_object](#)
- 1371. [xml_set_processing_instruction_handler](#)
- 1372. [xml_set_unparsed_entity_decl_handler](#)
- 1373. [xmldoc](#)
- 1374. [xmldocfile](#)
- 1375. [xmltree](#)

Y

- 1376. [yaz_addinfo](#)
- 1377. [yaz_close](#)
- 1378. [yaz_connect](#)
- 1379. [yaz_errno](#)
- 1380. [yaz_error](#)
- 1381. [yaz_hits](#)
- 1382. [yaz_range](#)
- 1383. [yaz_record](#)
- 1384. [yaz_search](#)
- 1385. [yaz_syntax](#)
- 1386. [yaz_wait](#)

- 1387. [yp_first](#)
- 1388. [yp_get_default_domain](#)
- 1389. [yp_master](#)
- 1390. [yp_match](#)
- 1391. [yp_next](#)
- 1392. [yp_order](#)

Z

- 1393. [zend_logo_guid](#)

Index des concepts

[Notes en ligne]

- 1394. , ,
- 1395. [établit le niveau d'erreur à prendre en compte.](#)
- 1396. [???, ???, ???, ???, ???, ???, ???, ???, ???](#)
- 1397. [@ref{function.strstr . , strstr\(\)}. insensible à la casse.](#)

A

- 1398. [Accepts a connection on a socket.](#)
- 1399. [Access CORBA objects](#)
- 1400. [Active l'approximation des translation d'objets.](#)
- 1401. [Active l'option décidant si, lors de la déconnexion du client, le script doit poursuivre son exécution ou non.](#)
- 1402. [Active la validation automatique.](#)
- 1403. [Active le debugger interne de PHP.](#)
- 1404. [Active ou désactive l'affichage des données de debuggage.](#)
- 1405. [Active ou désactive l'entrelacement.](#)
- 1406. [Active ou désactive le mode passif.](#)
- 1407. [Active/désactive l'option magic quotes runtime.](#)
- 1408. [Active/désactive le mode bloquant d'une socket.](#)
- 1409. [Addionne deux nombres de taille arbitraire.](#)
- 1410. [Adds a line to the history](#)
- 1411. [Affecte et/ou retourne l'identifiant de session courante](#)
- 1412. [Affecte et/ou retourne le chemin de sauvegarde de la session courante](#)
- 1413. [Affecte et/ou retourne le module courant de session courante](#)
- 1414. [Affecte et/ou retourne le nom de la session courante](#)
- 1415. [Affecte le champs créateur de la structure info.](#), [Affecte le champs créateur de la structure info.](#)
- 1416. [Affecte le champs mots-clé de la structure info.](#)
- 1417. [Affecte le champs titre de la structure info.](#)
- 1418. [Affecte le gestionnaire par défaut.](#)
- 1419. [Affecte les gestionnaires d'entité non déclaré. .](#)
- 1420. [Affecte les gestionnaires d'instructions exécutables. .](#)
- 1421. [Affecte les gestionnaires de caractère bruts.](#)
- 1422. [Affecte les gestionnaires de début et de fin.](#)
- 1423. [Affecte les gestionnaires de notation.](#)
- 1424. [Affecte les options d'un analyseur XML.](#)
- 1425. [Affecte un type à une variable.](#)
- 1426. [Affecte une nouvelle date de modification à un fichier.](#)
- 1427. [Affiche de nombreuses informations sur le PHP.](#)
- 1428. [Affiche des informations lisibles pour une variable.](#)
- 1429. [Affiche hw document.](#)
- 1430. [Affiche la partie du fichier située après le pointeur du fichier.](#)
- 1431. [Affiche le frame courant.](#)
- 1432. [Affiche le frame nommé.](#)
- 1433. [Affiche le résultat sous la forme d'une table HTML.](#)
- 1434. [Affiche ou affecte le paramètre "apache request notes".](#)

- 1435. [Affiche un fichier compressé](#)
- 1436. [Affiche un fichier.](#)
- 1437. [Affiche un message et termine le script courant](#)
- 1438. [Affiche un résultat sous forme de table HTML](#)
- 1439. [Affiche un texte à la position courante.](#)
- 1440. [Affiche un texte à une position donnée.](#)
- 1441. [Affiche un texte à une position.](#)
- 1442. [Affiche un texte dans un rectangle.](#)
- 1443. [Affiche un texte sur une nouvelle ligne.](#)
- 1444. [Affiche une chaîne formatée.](#)
- 1445. [Affiche une chaîne.](#)
- 1446. [Affiche une ou plusieurs chaînes.](#)
- 1447. [Ajoute des slashes dans une chaîne, comme en langage C.](#)
- 1448. [Ajoute un backslash devant tous les caractères méta](#)
- 1449. [Ajoute un enregistrement dans une base dBase.](#)
- 1450. [Ajoute un nouvel utilisateur virtuel, avec mot de passe](#), [Ajoute un nouvel utilisateur virtuel, avec mot de passe](#)
- 1451. [Ajoute un signet à la page courante.](#)
- 1452. [Ajoute un signet sur la page courante.](#)
- 1453. [Ajoute un slash devant tous les caractères spéciaux.](#)
- 1454. [Ajoute une annotation.](#), [Ajoute une annotation.](#)
- 1455. [Ajoute une chaîne dans une boîte au lettre.](#)
- 1456. [Ajoute une entrée à un dossier LDAP.](#)
- 1457. [Ajoute une valeur aux attributs courants.](#)
- 1458. [Ajouter des variables à un paquet WDDX.](#)
- 1459. [Aligne les dessins sur le chemin courant.](#)
- 1460. [Aligne sur le chemin courant.](#)
- 1461. [Alloue une couleur pour une image.](#)
- 1462. [Analyse un bloc binaire IPTC <http://www.xe.net/iptc/> et recherche les balises simples.](#)
- 1463. [Analyse un fichier en fonction d'un format](#)
- 1464. [Analyse une chaîne d'adresse.](#)
- 1465. [Analyse une chaîne, et en déduit des variables et leur valeur.](#)
- 1466. [Analyse une requête SQL.](#)
- 1467. [Analyse une requête.](#)
- 1468. [Analyse une URL et retourne ses composants.](#)
- 1469. [Annule les transactions en cours](#)
- 1470. [Annule une transaction](#), [Annule une transaction](#)
- 1471. [Annule une transaction.](#)
- 1472. [Applique une correction gamma à l'image](#)
- 1473. [arc cosinus](#)
- 1474. [arc sinus](#)
- 1475. [arc tangent](#)
- 1476. [arc tangent de deux variables](#)
- 1477. [Arrête l'animation flash.](#)
- 1478. [Arrondi à l'entier inférieur](#)
- 1479. [arrondi au nombre supérieur](#)
- 1480. [Arrondi.](#)
- 1481. [Attributs d'un objet.](#)
- 1482. [Attributs de l'objet dans l'ancrage.](#)
- 1483. [Attributs des ancrages d'un document.](#)
- 1484. [Attributs des documents fils d'un groupe.](#)

- 1485. [Attributs des parents.](#)
- 1486. [Authentification d'une connexion FTP](#)
- 1487. [Avance d'un frame.](#)
- 1488. [Avance le pointeur interne d'un tableau](#)

B

- 1489. [Balise de corps d'un document.](#)
- 1490. [Binds a name to a socket.](#)

C

- 1491. [Calcule l'intersection de tableaux](#)
- 1492. [Calcule la clé metaphone d'une chaîne.](#)
- 1493. [Calcule la différence entre deux tableaux](#)
- 1494. [Calcule la similarité de deux chaînes.](#)
- 1495. [Calcule la valeur de hash demandée par EZMLM](#)
- 1496. [Calcule la valeur soundex d'une chaîne.](#)
- 1497. [Calcule le nombre de couleur d'une palette.](#)
- 1498. [Calcule un hash.](#)
- 1499. [Calcule un md5 avec la chaîne.](#)
- 1500. [Call a user function given by the first parameter](#)
- 1501. [Change de dossier](#)
- 1502. [Change de dossier, et passe au dossier parent.](#)
- 1503. [Change l'espacement des caractères.](#)
- 1504. [Change l'inclinaison de la police courante.](#)
- 1505. [Change la couleur dans une palette à l'index donné.](#)
- 1506. [Change la police courante.](#)
- 1507. [Change la position courante.](#)
- 1508. [Change la taille de la police.](#)
- 1509. [Change le "umask" courant.](#)
- 1510. [Change le codage vectoriel d'un caractère dans une police.](#)
- 1511. [Change le dossier courant.](#)
- 1512. [Change le groupe possesseur du fichier.](#)
- 1513. [Change le groupe propriétaire du fichier.](#)
- 1514. [Change le mode du fichier.](#)
- 1515. [Change le mot de passe d'utilisateurs virtuels.](#)
- 1516. [Change le nom de session de l'utilisateur actif.](#)
- 1517. [Change les informations locales.](#)
- 1518. [Change spellchecking mode](#)
- 1519. [Charge un fichier ouvert sur un serveur FTP.](#)
- 1520. [Charge un fichier sur un serveur FTP.](#)
- 1521. [Charge un nouveau dictionnaire](#)
- 1522. [charge une extension PHP à la volée](#)
- 1523. [Charge une nouvelle police.](#)
- 1524. [Charge une police PostScript Type 1 depuis un fichier.](#)
- 1525. [Check a word](#)
- 1526. [Chemin du dossier courant.](#)
- 1527. [Choisi l'échelle.](#)
- 1528. [Choisi l'élévation du texte.](#)

- 1529. [Choisi l'origine du système de coordonnées.](#)
- 1530. [Choisi la "miter limit".](#)
- 1531. [Choisi la couleur grise comme couleur de remplissage et de dessin.](#)
- 1532. [Choisi la couleur grise comme couleur de remplissage.](#)
- 1533. [Choisi la couleur rgb comme couleur de dessin et de remplissage.](#)
- 1534. [Choisi la couleur rgb comme couleur de remplissage.](#), [Choisi la couleur rgb comme couleur de remplissage.](#)
- 1535. [Choisi la distance entre les lignes du textes.](#)
- 1536. [Choisi la durée de transition entre deux pages.](#)
- 1537. [Choisi la largeur de ligne.](#)
- 1538. [Choisi la rotation.](#)
- 1539. [Choisi le mode de remplissage par texture répétée.](#)
- 1540. [Choisi le mode de remplissage par texture.](#)
- 1541. [Choisi le mode par défaut de lecture des valeurs.](#)
- 1542. [Choisi le mode par défaut des objets BLOB pour toutes les requêtes SELECT.](#)
- 1543. [Choisi le mode par défaut des objets BYTE.](#)
- 1544. [Choisi le mode par défaut des objets text.](#)
- 1545. [Choisi le niveau de qualité haut ou bas.](#)
- 1546. [Choisi le paramètre linecap.](#)
- 1547. [Choisi le paramètre linejoin.](#)
- 1548. [Choisi le point courant.](#)
- 1549. [Choisi les caractères de remplissage.](#)
- 1550. [Choisit un niveau de gris comme couleur de dessin et de remplissage.](#)
- 1551. [Choisit un niveau de gris comme couleur de dessin.](#)
- 1552. [Choisit un niveau de gris comme couleur de remplissage.](#)
- 1553. [Choisit une couleur rgb comme couleur de dessin et de remplissage.](#)
- 1554. [Choisit une couleur rgb comme couleur de dessin.](#)
- 1555. [Choisit une couleur rgb comme couleur de remplissage.](#)
- 1556. [Classe dossier](#)
- 1557. [Clean \(erase\) the output buffer and turn off output buffering](#)
- 1558. [Clears the history](#)
- 1559. [Clos un paquet WDDX.](#)
- 1560. [Close a CURL session](#)
- 1561. [Close shared memory block](#)
- 1562. [Closes a YAZ connection](#)
- 1563. [Combine plusieurs tableaux ensemble, récursivement](#)
- 1564. [Commence l'analyse d'un fichier XML.](#)
- 1565. [Commence la définition d'un bouton.](#)
- 1566. [Commence la description d'une liste d'action pour la frame courante.](#)
- 1567. [Commence une forme complexe.](#)
- 1568. [Commence une nouvelle page.](#), [Commence une nouvelle page.](#)
- 1569. [Commencer un nouveau paquet WDDX avec une structure](#)
- 1570. [Compacte des données dans une chaîne binaire.](#)
- 1571. [Compacte une base dBase.](#)
- 1572. [Comparaison binaire de chaînes, insensible à la casse.](#)
- 1573. [Comparaison binaire de chaînes.](#)
- 1574. [Compare deux dates.](#)
- 1575. [Compare deux nombres de taille arbitraire.](#)
- 1576. [Complète la définition de la forme courante.](#)
- 1577. [Complète un tableau jusqu'à la longueur spécifiée, avec une valeur.](#)
- 1578. [Compte le nombre d'élément d'un tableau](#)

- 1579. [Compte le nombre d'entrées d'une recherche.](#)
- 1580. [Compte le nombre de champs d'une base dBase.](#)
- 1581. [Compte le nombre de ligne déjà lues dans un résultat.](#)
- 1582. [Compte le nombre de valeurs dans un tableau](#)
- 1583. [Compter le nombre d'enregistrements dans une base dBase.](#)
- 1584. [Connection persistante à un serveur Oracle.](#)
- 1585. [Connexion à un serveur](#)
- 1586. [Connexion à une source](#)
- 1587. [Consider run-together words as legal compounds](#)
- 1588. [Contenu d'un document.](#)
- 1589. [Contrôle la situation, l'aparance et la zone active du bouton courant.](#)
- 1590. [Converti d'octal en décimal.](#)
- 1591. [Converti de binaire en décimal](#)
- 1592. [Converti de décimal en binaire](#)
- 1593. [Converti de décimal en hexadécimal](#)
- 1594. [converti de décimal en octal](#)
- 1595. [Converti de hexadécimal en décimal](#)
- 1596. [Converti la chaîne d'un alphabet cyrillique vers un autre.](#)
- 1597. [Converti le nombre de jour du calendrier Julien en date du calendrier Julien.](#)
- 1598. [Converti le nombre de jours du calendrier julien en date du calendrier français républicain](#)
- 1599. [Converti le nombre de jours du calendrier julien en date du calendrier juif.](#)
- 1600. [Converti le nombre de jours du calendrier Julien en date du calendrier Julien.](#)
- 1601. [Converti le nombre de jours du calendrier Julien en date grégorienne.](#)
- 1602. [Converti les attributs d'un objet en tableau.](#)
- 1603. [Converti les nouvelles lignes en HTML \(<BR\).](#)
- 1604. [Converti tous les caractères spéciaux en équivalent HTML.](#), [Converti tous les caractères spéciaux en équivalent HTML.](#)
- 1605. [Converti un ND dans un format plus accessible.](#)
- 1606. [Converti un nombre en des bases arbitraires.](#)
- 1607. [Converti un numéro d'erreur LDAP en message d'erreur.](#)
- 1608. [converti un tableau en un objet.](#)
- 1609. [Converti un une chaîne UTF-8 en ISO-8859.](#)
- 1610. [Converti une adresse IP \(IPv4\) en adresse IP numérique](#)
- 1611. [Converti une chaîne à 8 bits en une chaîne à base64.](#)
- 1612. [Converti une chaîne contenant une adresse \(IPv4\) IP numérique en adresse littérale.](#)
- 1613. [Converti une date du calendrier français républicain en nombre de jours du calendrier julien.](#)
- 1614. [Converti une date du calendrier juif en nombre de jours du calendrier julien.](#)
- 1615. [Converti une date grégorienne en nombre de jours du calendrier julien.](#)
- 1616. [Converti une valeur binaire en hexadécimal](#)
- 1617. [Convertie des données 8bit en texte UTF-7.](#)
- 1618. [Convertit un une chaîne ISO-8859-1 en UTF-8.](#)
- 1619. [Convertit une chaîne à 8 bits en une chaîne à guillemets.](#)
- 1620. [Convertit une chaîne à guillemets en une chaîne à 8 bits.](#)
- 1621. [Copie des objets.](#)
- 1622. [Copie et redimensionne une partie d'une image.](#)
- 1623. [Copie les messages spécifiés dans une boîte aux lettres.](#)
- 1624. [Copie un fichier.](#)
- 1625. [Copie une partie d'une image](#)
- 1626. [cosinus](#)
- 1627. [Création d'un analyseur XML.](#)
- 1628. [Crée ou ouvre un segment de mémoire partagée.](#)

- 1629. [Crée un arbre d'objet PHP, à partir d'un document XML.](#)
- 1630. [Crée un dossier.](#), [Crée un dossier.](#)
- 1631. [Crée un fichier avec un nom unique.](#)
- 1632. [Crée un fichier fifo \(first in, first out\) \(un pipe nommé\) .](#)
- 1633. [Crée un fichier temporaire](#)
- 1634. [Crée un lien symbolique.](#)
- 1635. [Crée un lien.](#)
- 1636. [Crée un nouveau document FDF.](#)
- 1637. [Crée un nouveau document.](#)
- 1638. [Crée un objet BLOB.](#)
- 1639. [Crée un objet char.](#)
- 1640. [Crée un objet de grande taille.](#)
- 1641. [Crée un objet DOM à partir d'un fichier XML](#)
- 1642. [Crée un objet DOM pour un document XML.](#)
- 1643. [Crée un objet SLOB et l'ouvre.](#)
- 1644. [Crée un processus de pointeur de fichier.](#)
- 1645. [Crée un tableau](#)
- 1646. [Crée un tableau contenant les variables et leur valeur](#)
- 1647. [Crée un tableau contenant un intervalle d'entiers](#)
- 1648. [Crée un vecteur d'initialisation à partir d'une source aléatoire.](#)
- 1649. [Crée une base de données dBase.](#)
- 1650. [Crée une base de données mSQL.](#), [Crée une base de données mSQL.](#)
- 1651. [Crée une base de données MySQL.](#)
- 1652. [Crée une connexion persistante.](#)
- 1653. [Crée une nouvelle boîte aux lettres.](#)
- 1654. [Crée une nouvelle image à partir d'un fichier ou d'une URL *JPEG*](#)
- 1655. [Crée une nouvelle image à partir d'un fichier ou d'une URL *PNG*](#)
- 1656. [Crée une nouvelle image à partir d'un fichier ou d'une URL.](#)
- 1657. [Crée une nouvelle image.](#)
- 1658. [Create a socket \(endpoint for communication\).](#)
- 1659. [Create an anonymous \(lambda-style\) function](#)
- 1660. [Create or open shared memory block](#)

D

- 1661. [Décode les données de session à partir d'une chaîne](#)
- 1662. [Décode un texte encodé en BASE64.](#)
- 1663. [Décode une chaîne](#)
- 1664. [Décode une chaîne en MIME base64](#)
- 1665. [Décode une chaîne encodée URL.](#)
- 1666. [Décode une chaîne modifiée UTF-7.](#)
- 1667. [Décode une chaîne URL.](#)
- 1668. [Déconditionne des données depuis une chaîne binaire.](#)
- 1669. [Déconnecte d'un serveur LDAP.](#)
- 1670. [Déconnection d'un serveur Oracle](#)
- 1671. [Décrit la librairie dbm utilisée.](#)
- 1672. [Décrit une transition utilisée pour déclencher une liste d'actions.](#)
- 1673. [Décrypte](#)
- 1674. [Décrypte un texte](#)
- 1675. [Dédoublonne un tableau](#)

- 1676. [Défini le point de vue de l'utilisateur en coordonnées polaire.](#)
- 1677. [Définit la couleur transparente.](#)
- 1678. [Définit un polygone.](#)
- 1679. [Définit un rectangle.](#)
- 1680. [Définit un symbole.](#)
- 1681. [Définit une chaîne de texte.](#)
- 1682. [Définit une constante.](#)
- 1683. [Définit une image bitmap.](#)
- 1684. [Définit une ligne.](#)
- 1685. [Définit une police.](#)
- 1686. [Définit une projection orthogonale à 2 dimensions entre les coordonnées utilisateur et le port courant.](#)
- 1687. [définit une projection orthogonale à 3 dimensions entre les coordonnées utilisateur et le port courant](#)
- 1688. [Définit une projection orthogonale entre les coordonnées utilisateur et le port courant.](#)
- 1689. [Définit une transformation de vue.](#)
- 1690. [Démarré une section de texte.](#)
- 1691. [Dépile la matrice de transformation.](#)
- 1692. [Dépile un élément au début d'un tableau](#)
- 1693. [Dépile un élément de la fin d'un tableau](#)
- 1694. [Déplace le pointeur courant dans un fichier compressé](#)
- 1695. [Déplace le pointeur interne de ligne.](#)
- 1696. [Déplace le pointeur interne de lignes.](#)
- 1697. [Déplace le pointeur interne de résultat.](#)
- 1698. [Déplace le pointeur interne.](#)
- 1699. [Déplace les messages spécifiés dans une boîte aux lettres.](#)
- 1700. [Déplace un fichier téléchargé.](#)
- 1701. [Déplace un objet.](#)
- 1702. [Déquote une chaîne quotée avec addcslashes](#)
- 1703. [Désaffecte une variable.](#)
- 1704. [Désalloune une couleur pour une image](#)
- 1705. [Détermine le nombre de décimales par défaut pour les fonctions de précision mathématiques.](#)
- 1706. [détermine si une extension est chargée ou non.](#)
- 1707. [Détermine si une variable est affectée., Détermine si une variable est affectée.](#)
- 1708. [Détermine si une variable est de type double.](#)
- 1709. [Détermine si une variable est de type float.](#)
- 1710. [Détermine si une variable est de type int.](#)
- 1711. [Détermine si une variable est de type integer., Détermine si une variable est de type integer.](#)
- 1712. [Détermine si une variable est de type object.](#)
- 1713. [Détermine si une variable est de type real.](#)
- 1714. [Détermine si une variable est de type string.](#)
- 1715. [Détermine si une variable est un tableau.](#)
- 1716. [Détermine le rendu du texte.](#)
- 1717. [Détruit toutes les données enregistrées d'une session](#)
- 1718. [Détruit un analyseur XML.](#)
- 1719. [Détruit un document.](#)
- 1720. [détruit une image.](#)
- 1721. [Déverrouille un objet.](#)
- 1722. [Defface un enregistrement dans une base dBase.](#)
- 1723. [Delete shared memory block](#)
- 1724. [Demande d'informations d'arbre sur une entité du réseau.](#)
- 1725. [Dessine le long du chemin.](#)
- 1726. [Dessine un arc de cercle.](#)

- 1727. [Dessine un arc.](#)
- 1728. [Dessine un caractère horizontalement.](#)
- 1729. [Dessine un caractère verticalement.](#)
- 1730. [Dessine un cercle., Dessine un cercle.](#)
- 1731. [Dessine un pixel.](#)
- 1732. [Dessine un polygone rempli.](#)
- 1733. [Dessine un polygone.](#)
- 1734. [Dessine un rectangle rempli.](#)
- 1735. [Dessine un rectangle., Dessine un rectangle., Dessine un rectangle.](#)
- 1736. [Dessine un texte avec une police TrueType.](#)
- 1737. [Dessine un texte sur une image avec une police PostScript Type 1.](#)
- 1738. [Dessine une arc de cercle.](#)
- 1739. [Dessine une chaîne horizontale.](#)
- 1740. [Dessine une chaîne verticale.](#)
- 1741. [Dessine une courbe Bézier cubique.](#)
- 1742. [Dessine une courbe de Bézier quadratique entre deux points.](#)
- 1743. [Dessine une courbe., Dessine une courbe.](#)
- 1744. [Dessine une ellipse partielle.](#)
- 1745. [Dessine une ligne le long du chemin.](#)
- 1746. [Dessine une ligne pointillée.](#)
- 1747. [Dessine une ligne, relativement.](#)
- 1748. [Dessine une ligne., Dessine une ligne., Dessine une ligne.](#)
- 1749. [Determine le rendu du texte.](#)
- 1750. [Determine si un pointeur de fichier est un terminal interactif .](#)
- 1751. [Divise deux nombres de taille arbitraire.](#)
- 1752. [Draw a line.](#)
- 1753. [Dumpe les informations d'une variable.](#)
- 1754. [Duplique un objet BLOB.](#)

E

- 1755. [Echappe les méta-caractères Shell.](#)
- 1756. [Echappement des caractères spéciaux des expressions régulières.](#)
- 1757. [Eclatement d'une chaîne par expression régulière.](#)
- 1758. [Ecrit dans un fichier compressé](#)
- 1759. [Ecrit dans un fichier.](#)
- 1760. [Ecrit la valeur courante de l'option quick_print de la librairie UCD.](#)
- 1761. [Ecrit un document PDF dans un fichier.](#)
- 1762. [Ecrit un fichier compressé en mode binaire](#)
- 1763. [Ecrit un objet de grande taille](#)
- 1764. [Ecrit une chaîne dans un objet SLOB.](#)
- 1765. [Ecriture du fichier en mode binaire.](#)
- 1766. [Efface des objets.](#)
- 1767. [Efface et remplace une portion de tableau](#)
- 1768. [Efface le cache de la fonction "stat".](#)
- 1769. [Efface le résultat de la mémoire.](#)
- 1770. [Efface tous les messages marqués pour l'effacement.](#)
- 1771. [Efface un événement dans un calendrier MCAL.](#)
- 1772. [Efface un événement dans un agenda ICAP.](#)
- 1773. [Efface un dossier., Efface un dossier.](#)

- 1774. [Efface un fichier sur un serveur FTP.](#)
- 1775. [Efface un fichier.](#)
- 1776. [Efface un objet de grande taille](#)
- 1777. [Efface une base de données mSQL.](#), [Efface une base de données mSQL.](#)
- 1778. [Efface une base de données MySQL.](#)
- 1779. [Efface une boîte aux lettres.](#)
- 1780. [Efface une entrée dans un dossier.](#)
- 1781. [Efface une entrée.](#)
- 1782. [Efface une valeur des attributs courants.](#)
- 1783. [Efface une valeur.](#)
- 1784. [Efface une variable de la mémoire partagée.](#)
- 1785. [Effacer](#)
- 1786. [Effectue une requête partielle pour l'URI spécifiée et renvoie toutes les informations.](#)
- 1787. [Effectue une rotation.](#)
- 1788. [Effectue une sous-requête Apache](#)
- 1789. [Elève un nombre à la puissance n-ième.](#)
- 1790. [Empile la matrice de transformation courante dans la pile.](#)
- 1791. [Empile un ou plusieurs éléments à la fin d'un tableau](#)
- 1792. [Empile un ou plusieurs éléments au début d'un tableau](#)
- 1793. [Encode les données de session dans une chaîne](#)
- 1794. [Encode une chaîne en MIME base64.](#)
- 1795. [Encode une chaîne en URL, selon la RFC1738.](#)
- 1796. [Encode une chaîne en URL.](#)
- 1797. [Encrypte](#)
- 1798. [Encrypte un texte](#)
- 1799. [Encrypte une chaîne avec un DES.](#)
- 1800. [Encrypte/décrypte des données en mode CBC](#)
- 1801. [Encrypte/décrypte des données en mode CFB](#)
- 1802. [Encrypte/décrypte des données en mode ECB](#)
- 1803. [Encrypte/décrypte des données en mode OFB](#)
- 1804. [Enlève la marque d'effacement d'un message.](#)
- 1805. [Enlève les balises HTML et PHP.](#)
- 1806. [Enlève les espaces de début de chaîne.](#)
- 1807. [Enlève les espaces de fin de chaîne.](#)
- 1808. [Enlève les espaces de fin et de fin de chaîne.](#)
- 1809. [Enlève les slash ajoutés par la fonction addslashes](#)
- 1810. [Enlève un objet.](#)
- 1811. [Enregistre l'environnement courant.](#)
- 1812. [Enregistre un événement dans un agenda ICAP.](#)
- 1813. [Enregistre un nouvel événement dans un calendrier MCAL.](#)
- 1814. [Enregistre une fonction pour exécution à l'extinction.](#)
- 1815. [Enregistre une image dans un fichier PDF pour utilisation ultérieure.](#)
- 1816. [Enregistre une variable dans la session courante](#)
- 1817. [Enregistrer plusieurs valeurs dans un paquet WDDX](#)
- 1818. [Enregistrer une valeur dans un paquet WDDX](#)
- 1819. [Envoie la commande SITE au serveur.](#)
- 1820. [Envoie le document PDF dans un buffer mémoire.](#)
- 1821. [Envoie un cookie](#)
- 1822. [Envoie un entête HTTP.](#)
- 1823. [Envoie un message d'erreur.](#)
- 1824. [Envoie un objet SNMP.](#)

- 1825. [Envoie un signal à un process.](#)
- 1826. [Envoie une image GIF vers un navigateur ou un fichier.](#)
- 1827. [Envoie une image JPEG vers un navigateur ou un fichier.](#)
- 1828. [Envoie une image PNG vers un navigateur ou un fichier.](#)
- 1829. [Envoie une requête à une base Sybase.](#)
- 1830. [Envoie une requête Informix.](#)
- 1831. [Envoie une requête mSQL.](#)
- 1832. [Envoie une requête MySQL à un serveur MySQL.](#)
- 1833. [Envoie une requête SQL à un serveur MySQL.](#)
- 1834. [Envoie une requête SQL.](#)
- 1835. [Etablit une connexion à un serveur Oracle.](#)
- 1836. [Etablit une connexion persistante.](#)
- 1837. [Eteind l'alarme d'un événement., Eteind l'alarme d'un événement.](#)
- 1838. [Etend ou condense une police de caractères](#)
- 1839. [Evalue une chaîne comme un script PHP.](#)
- 1840. [Exécute un programme externe et affiche le résultat brut.](#)
- 1841. [Exécute un programme externe et affiche le résultat.](#)
- 1842. [Exécute un programme externe.](#)
- 1843. [Exécute une commande](#)
- 1844. [Exécute une commande analysée sur un pointeur Oracle.](#)
- 1845. [Exécute une fonction sur chacun des membres d'un tableau.](#)
- 1846. [Exécute une requête mSQL.](#)
- 1847. [Exécute une requête préparée](#)
- 1848. [Exécute une requête préparée.](#)
- 1849. [Exécute une requête SQL déjà préparée.](#)
- 1850. [Exécute une requête SQL préparée.](#)
- 1851. [Exécute une requête sur une base Interbase](#)
- 1852. [Exécute une requête.](#)
- 1853. [Executes an SQL statement](#)
- 1854. [Executes queries](#)
- 1855. [exponentielle](#)
- 1856. [Expression régulière globale.](#)
- 1857. [Expression régulière standard., Expression régulière standard.](#)
- 1858. [Extrait toutes les balises meta d'un fichier, et les retourne sous forme d'un tableau.](#)
- 1859. [Extrait une portion de tableau](#)

F

- 1860. [Fait du processus courant un chef de session.](#)
- 1861. [Fait une liste détaillée de fichier dans un dossier.](#)
- 1862. [Ferme et clos le chemin.](#)
- 1863. [Ferme la connexion à l'historique système.](#)
- 1864. [Ferme la connexion Hyperwave.](#)
- 1865. [Ferme la connexion MySQL.](#)
- 1866. [Ferme le chemin courant.](#)
- 1867. [Ferme le chemin et dessine le long du chemin.](#)
- 1868. [Ferme le chemin.](#)
- 1869. [Ferme le fichier courant Shockwave Flash.](#)
- 1870. [Ferme le fichier et dessine une ligne le long du chemin.](#)
- 1871. [Ferme le flot ICAP.](#)

- 1872. [Ferme le pointeur sur le dossier.](#)
- 1873. [Ferme toutes les connexions aux serveur ovrimos](#)
- 1874. [Ferme toutes les connexions ODBC](#)
- 1875. [Ferme un document PDF.](#)
- 1876. [Ferme un fichier PDF.](#)
- 1877. [Ferme un fichier.](#)
- 1878. [Ferme un objet de grande taille.](#)
- 1879. [Ferme un objet SLOB.](#)
- 1880. [Ferme un pointeur Oracle.](#)
- 1881. [Ferme un pointeur sur un fichier compressé.](#)
- 1882. [Ferme un processus de pointeur de fichier.](#)
- 1883. [Ferme une base dBase.](#)
- 1884. [Ferme une base de données dbm.](#)
- 1885. [Ferme une base.](#)
- 1886. [Ferme une connexion](#)
- 1887. [Ferme une connexion à un serveur Informix.](#)
- 1888. [Ferme une connexion à une base de données Interbase.](#)
- 1889. [Ferme une connexion FTP.](#)
- 1890. [Ferme une connexion MCAL.](#)
- 1891. [Ferme une connexion MS SQL Server.](#)
- 1892. [Ferme une connexion mSQL.](#)
- 1893. [Ferme une connexion ODBC.](#)
- 1894. [Ferme une connexion Oracle.](#)
- 1895. [Ferme une connexion Sybase.](#)
- 1896. [Ferme une document FDF.](#)
- 1897. [Ferme une image.](#)
- 1898. [Fills the author field of the info structure.](#)
- 1899. [Fixe certains paramètres.](#)
- 1900. [Fixe d'offset d'un champs.](#)
- 1901. [Fixe l'échelle horizontale du texte.](#)
- 1902. [Fixe l'élévation du texte.](#)
- 1903. [Fixe l'alarme de la structure globale.](#)
- 1904. [Fixe l'animation de la transition entre les pages.](#)
- 1905. [Fixe l'apparence d'un champs.](#)
- 1906. [Fixe l'echelle horizontale du texte.](#)
- 1907. [Fixe l'espacement des caractères.,](#) [Fixe l'espacement des caractères.](#)
- 1908. [Fixe l'espacement des mots.,](#) [Fixe l'espacement des mots.](#)
- 1909. [Fixe l'identifiant de group de processus.](#)
- 1910. [Fixe l'offset du pointeur de champs.](#)
- 1911. [Fixe l'UID effective du processus courant.](#)
- 1912. [Fixe la bufferisation de fichier](#)
- 1913. [Fixe la catégorie de la structure globale.](#)
- 1914. [Fixe la classe de la structure globale.](#)
- 1915. [Fixe la couleur de dessin à un niveau de gris.](#)
- 1916. [Fixe la couleur globale d'addition \(? : the global add color\).](#)
- 1917. [Fixe la couleur globale de multiplication \(? : the global multiply color\).](#)
- 1918. [Fixe la couleur pour le style courant de remplissage.](#)
- 1919. [Fixe la description de la structure globale.](#)
- 1920. [Fixe la date de fin de la structure globale.](#)
- 1921. [Fixe la distance entre deux lignes.](#)
- 1922. [Fixe la durée de vie de la socket](#)

- 1923. [Fixe la largeur de ligne.](#)
- 1924. [Fixe la matrice de texte.](#)
- 1925. [Fixe la matrice du texte.](#)
- 1926. [Fixe la page courante.](#)
- 1927. [Fixe la platitude \(flatness\).](#), [Fixe la platitude \(flatness\).](#)
- 1928. [Fixe la position du texte.](#), [Fixe la position du texte.](#)
- 1929. [Fixe la prochaine ligne dans le pointeur interne de résultat.](#)
- 1930. [fixe la récurrence annuelle.](#)
- 1931. [Fixe la récurrence hebdomadaire.](#)
- 1932. [fixe la récurrence mensuelle.](#)
- 1933. [Fixe la récurrence quotidienne.](#)
- 1934. [Fixe la récurrence.](#)
- 1935. [Fixe la sévérité minimale du client](#)
- 1936. [Fixe la sévérité minimale du client pour le serveur](#)
- 1937. [Fixe la sévérité minimale du client pour les erreurs](#)
- 1938. [Fixe la sévérité minimale du client pour les messages](#)
- 1939. [Fixe la valeur d'un champs.](#)
- 1940. [Fixe la valeur d'une variable d'environnement.](#)
- 1941. [Fixe la valeur de la clé /F.](#)
- 1942. [Fixe la valeur de la clé /STATUS.](#)
- 1943. [Fixe le chemin d'un domaine.](#)
- 1944. [Fixe le contexte des actions.](#)
- 1945. [Fixe le créateur d'un document PDF.](#)
- 1946. [Fixe le domaine par défaut.](#)
- 1947. [Fixe le fichier courant, ou la position courante.](#)
- 1948. [Fixe le format de date pour les prochaines requêtes.](#)
- 1949. [Fixe le frame courant.](#)
- 1950. [Fixe le gestionnaire de référence externes.](#)
- 1951. [Fixe le GID effective du processus courant.](#)
- 1952. [Fixe le mode de transition entre les pages.](#)
- 1953. [Fixe le motif de pointillé.](#)
- 1954. [Fixe le niveau de sévérité des erreurs.](#)
- 1955. [Fixe le niveau de sévérité des messages d'erreurs.](#)
- 1956. [Fixe le paramètre linecap.](#)
- 1957. [Fixe le paramètre linejoin.](#)
- 1958. [Fixe le paramètre miter limit.](#)
- 1959. [Fixe le point courant relativement.](#)
- 1960. [Fixe le point courant.](#)
- 1961. [Fixe le style courant de ligne.](#)
- 1962. [Fixe le sujet d'un document PDF.](#)
- 1963. [Fixe le temps maximum d'exécution d'un script.](#)
- 1964. [Fixe le titre d'un document PDF.](#)
- 1965. [fixe le titre de la structure globale.](#)
- 1966. [Fixe les dates de début et de fin de la structure globale.](#)
- 1967. [Fixe les limites d'un document PDF.](#)
- 1968. [Fixe les mot clés d'un document PDF.](#)
- 1969. [Flush \(send\) the output buffer and turn off output buffering](#)
- 1970. [Fonctionnement des expressions régulières.](#)
- 1971. [Force le premier caractère d'une chaîne en majuscule.](#)
- 1972. [Force le premier caractère de chaque mot d'une chaîne en majuscule](#)
- 1973. [Formate un nombre par groupe de milliers.](#)

- 1974. [Formate une date/heure GMT/CUT en fonction des paramètres locaux.](#)
- 1975. [Formate une date/heure GMT/CUT.](#)
- 1976. [Formate une date/heure locale](#)
- 1977. [Formate une date/heure locale avec les options locales.](#)
- 1978. [Fuite de mémoire.](#)

G

- 1979. [Génère un identifiant unique.](#)
- 1980. [Génère un message dans l'historique système.](#)
- 1981. [Génère une meilleure valeur aléatoire.](#)
- 1982. [Génère une représentation enregistrable d'une valeur.](#)
- 1983. [Génère une valeur aléatoire.](#)
- 1984. [Génère une clé](#)
- 1985. [Generates a user-level error/warning/notice message.](#) [Generates a user-level error/warning/notice message](#)
- 1986. [Gestion des colonnes de données binaires.](#)
- 1987. [Gestion des colonnes de type LONG.](#)
- 1988. [Get result data.](#)
- 1989. [Get size of shared memory block](#)
- 1990. [Get the exception struct for the latest exception](#)
- 1991. [Get the repository id for the latest exception.](#)
- 1992. [Gets/sets various internal readline variables](#)

H

- 1993. [Homothétie.](#)

I

- 1994. [Identifiant d'objet de l'objet dans l'ancrage.](#)
- 1995. [Identifiant d'objet des groupes fils.](#)
- 1996. [Identifiant d'objet des parents.](#)
- 1997. [Identifiants des ancrages d'un document.](#)
- 1998. [Identifie un utilisateur.](#)
- 1999. [ids des documents fils d'un groupe.](#)
- 2000. [Ignore les actions si le frame n'est pas chargé.](#)
- 2001. [Import variables into the symbol table from an array](#)
- 2002. [Imprime le texte à la ligne suivante.](#)
- 2003. [Imprime un texte à la position courante.](#)
- 2004. [Imprime un texte avec des options.](#)
- 2005. [Inactive la validation automatique.](#)
- 2006. [Inactive le debugger interne de PHP.](#)
- 2007. [Inactive le remplissage.](#)
- 2008. [Includnet une police de caractères](#)
- 2009. [Indique de quoi est capable le navigateur client.](#)
- 2010. [Indique la taille des données à lire dans une colonne de grande taille](#)
- 2011. [Indique les tailles de clé supportées par un algorithme](#)
- 2012. [Indique si le fichier a été téléchargé par HTTP POST](#)

- 2013. [Indique si le fichier est exécutable.](#)
- 2014. [Indique si le fichier est un lien symbolique.](#)
- 2015. [Indique si le fichier est un véritable fichier.](#)
- 2016. [Indique si le nom de fichier est un dossier.](#)
- 2017. [Indique si un algorithme fonctionne par bloc](#)
- 2018. [Indique si un mode fonctionne par bloc](#)
- 2019. [Indique si un mode travaille par blocs](#)
- 2020. [Indique si une valeur existe.](#)
- 2021. [Indique si une variable a été enregistrée dans la session ou pas](#)
- 2022. [indique un fichier est autorisé en lecture](#)
- 2023. [Indique un fichier est autorisé en lecture.](#)
- 2024. [Informations à propos d'une connexion.](#)
- 2025. [Initialise la structure globale d'un flot.](#)
- 2026. [Initialise le générateur de nombres aléatoires](#)
- 2027. [Initialise les données de session](#)
- 2028. [Initialise tous les buffers nécessaires](#)
- 2029. [Initialise toutes les constantes liées au syslog](#)
- 2030. [Initialise un nouveau pointeur vide de LOB/FILE](#)
- 2031. [Initialise une meilleure valeur aléatoire](#)
- 2032. [Initialises the Payflow Pro library](#)
- 2033. [Initialize a CURL session](#)
- 2034. [Initiates a connection on a socket.](#)
- 2035. [Insère ou modifie une variable de la mémoire partagée.](#)
- 2036. [Insère un document dans un groupe.](#)
- 2037. [Insère un document.](#)
- 2038. [Insère un groupe.](#)
- 2039. [Insère un objet record.](#)
- 2040. [Insère une entrée.](#)
- 2041. [Insère une valeur.](#)
- 2042. [Inverse l'ordre des caractères d'une chaîne.](#)

J

- 2043. [Joue l'animation flash à partir du frame courant.](#)
- 2044. [Joue un frame puis stoppe.](#)

L

- 2045. [La plus grand valeur aléatoire possible.](#)
- 2046. [La plus grande valeur.](#)
- 2047. [La plus petite valeur.](#)
- 2048. [Le jour de l'année.](#)
- 2049. [Le jour de la semaine.](#)
- 2050. [Lecture du fichier en mode binaire.](#)
- 2051. [Lecture du pointeur de fiche courante \(cursorname\).](#)
- 2052. [Libère la mémoire](#)
- 2053. [Libère la mémoire occupée par une police PostScript Type 1.](#)
- 2054. [Libère la mémoire prise par un résultat.](#)
- 2055. [Libère la mémoire réservée par une requête préparée.](#)
- 2056. [Libère la mémoire.](#)

- 2057. [Libère le résultat de la mémoire.](#), [Libère le résultat de la mémoire.](#)
- 2058. [Libère les ressources associées à un résultat](#)
- 2059. [Libère les ressources prises par un résultat.](#)
- 2060. [Libère les ressources utilisées par un résultat](#)
- 2061. [Libère toutes les ressources occupées par un pointeur.](#)
- 2062. [Libère toutes les ressources occupées par une commande.](#)
- 2063. [Libère un résultat de la mémoire.](#)
- 2064. [Libère un résultat.](#)
- 2065. [Libère un sémaphore.](#)
- 2066. [Lie les paramètres avec une requête précédemment préparée.](#)
- 2067. [Lie une variable PHP à un paramètre Oracle.](#)
- 2068. [Lire le nom du groupe](#)
- 2069. [Lire un paquet WDDX.](#)
- 2070. [Lis la liste des boîtes aux lettres, et y recherche une chaîne.](#)
- 2071. [Lis le corps d'un message.](#)
- 2072. [Lis les entêtes EXIF d'une image JPEG](#)
- 2073. [Lis les informations à propos de la boîte aux lettres courante.](#)
- 2074. [Liste des object ids des objets fils.](#)
- 2075. [Liste des object records des objets fils.](#)
- 2076. [Liste des utilisateurs actuellement identifiés.](#)
- 2077. [Liste les bases de données disponibles sur le serveur MySQL.](#)
- 2078. [Liste les bases de données mSQL sur un serveur.](#), [Liste les bases de données mSQL sur un serveur.](#)
- 2079. [Liste les boîtes aux lettres souscrites.](#)
- 2080. [Liste les boîtes aux lettres, et retourne le détail pour chacune.](#)
- 2081. [Liste les boîtes aux lettres.](#)
- 2082. [Liste les champs d'une table.](#), [Liste les champs d'une table.](#)
- 2083. [Liste les champs du résultat MySQL.](#)
- 2084. [Liste les champs Informix SQL.](#)
- 2085. [Liste les propriétés des champs SQL.](#)
- 2086. [Liste les tables d'une base de données.](#)
- 2087. [Liste les tables mSQL sur une base de données](#)
- 2088. [Liste les tables mSQL sur une base de données.](#)
- 2089. [Liste tous les algorithmes de chiffrement supportés](#)
- 2090. [Liste tous les modes de chiffrement supportés](#)
- 2091. [Liste toutes les boîtes aux lettres souscrites.](#)
- 2092. [Listens for a connection on a socket.](#)
- 2093. [Lists the history](#)
- 2094. [Lit et vérifie un fichier.](#)
- 2095. [Lit l'attribut suivant.](#), [Lit l'attribut suivant.](#)
- 2096. [Lit l'entête d'un message.](#)
- 2097. [Lit la clé suivante.](#), [Lit la clé suivante.](#)
- 2098. [Lit la longueur d'un champs.](#), [Lit la longueur d'un champs.](#)
- 2099. [Lit la première clé.](#), [Lit la première clé.](#)
- 2100. [Lit la structure d'un message.](#)
- 2101. [Lit la totalité d'un fichier compressé dans un tableau.](#)
- 2102. [Lit la valeur courante de l'option quick_print de la librairie UCD.](#)
- 2103. [Lit la valeur d'un champs.](#)
- 2104. [Lit la valeur de la clé /F.](#)
- 2105. [Lit la valeur de la clé /STATUS.](#)
- 2106. [Lit le contenu d'une colonne](#)
- 2107. [Lit le fichier et renvoie le résultat dans un tableau.](#)

- 2108. [Lit le message d'erreur de l'analyseur XML.](#)
- 2109. [Lit le nom d'un champs.](#), [Lit le nom d'un champs.](#), [Lit le nom d'un champs.](#)
- 2110. [Lit le nom de la base de données courante.](#)
- 2111. [Lit le nom de la colonne.](#)
- 2112. [Lit le nom du champs suivant.](#)
- 2113. [Lit le nom du chiffrement utilisé.](#)
- 2114. [Lit les données de résultat.](#)
- 2115. [Lit les données liées à une clé.](#)
- 2116. [Lit les inforamtions sur le champs.](#)
- 2117. [Lit les informations d'un champs.](#)
- 2118. [Lit les informations sur un fichier à partir d'un pointeur de fichier](#)
- 2119. [Lit les informations sur une image.](#)
- 2120. [Lit les options d'un analyseur XML.](#)
- 2121. [Lit n bytes d'un objet SLOB.](#)
- 2122. [Lit toutes les informations restantes d'un fichier compressé](#)
- 2123. [Lit toutes les lignes d'un tableau, et la met sous la forme d'un tableau HTML.](#)
- 2124. [Lit un caractère d'un fichier compressé.](#)
- 2125. [Lit un enregistrement dans une base dBase.](#)
- 2126. [Lit un enregistrement dans une base, sous la forme d'un tableau associatif.](#)
- 2127. [Lit un fichier compressé en mode binaire](#)
- 2128. [Lit un objet de grande taille en totalité.](#)
- 2129. [Lit un objet de grande taille.](#)
- 2130. [Lit une entrée du dossier.](#)
- 2131. [Lit une entrée.](#)
- 2132. [Lit une ligne comme un tableau.](#)
- 2133. [Lit une ligne d'un fichier compressé](#)
- 2134. [Lit une ligne d'un fichier compressé et supprime les balises HTML](#)
- 2135. [Lit une ligne dans un objet.](#)
- 2136. [Lit une ligne dans un résultat,](#) [Lit une ligne dans un résultat](#)
- 2137. [Lit une ligne dans un tableau.,](#) [Lit une ligne dans un tableau.,](#) [Lit une ligne dans un tableau.](#)
- 2138. [Lit une ligne dans une base Interbase](#)
- 2139. [Lit une ligne dans une base Interbase dans un objet.](#)
- 2140. [Lit une ligne de résultat, et la place dans un tableau.](#)
- 2141. [Lit une ligne de résultat.](#)
- 2142. [Lit une ligne sous la forme d'un objet.](#)
- 2143. [Lit une ligne sous la forme d'un tableau.](#)
- 2144. [Lit une représentation enregistrée d'une valeur.](#)
- 2145. [Lit une valeur dans un résultat.](#)
- 2146. [Lit une valeur.](#)
- 2147. [Lit une variable dans la mémoire partagée.](#)
- 2148. [Load a new dictionary](#)
- 2149. [logarithme en base 10.](#)
- 2150. [Logarithme naturel](#)

M

- 2151. [Mélange les éléments d'un tableau](#)
- 2152. [Marque le fichier pour l'effacement, dans la boîte aux lettres courante.](#)
- 2153. [Message d'erreur.](#)
- 2154. [Met tous les caractères en majuscule.](#)

- 2155. [Met tous les caractères en minuscule.](#)
- 2156. [Mode auto-validation](#)
- 2157. [Modifie l'echelle.](#)
- 2158. [Modifie l'index d'un champs.](#)
- 2159. [Modifie l'origine du système de coordonnées.](#)
- 2160. [Modifie le contenu d'un objet BLOB.](#)
- 2161. [Modifie le contenu d'un objet char.](#)
- 2162. [Modifie le pointeur de fichier.](#)
- 2163. [Modifie le système de coordonnées.](#)
- 2164. [Modifie les attributs d'objet record.](#)
- 2165. [Modifie les paramètres ODBC.](#)
- 2166. [Modifie un événement dans un calendrier MCAL.](#)
- 2167. [Modifie un objet.](#)
- 2168. [Modifie une entrée LDAP.](#)
- 2169. [Modifie/remplace le contenu d'un document.](#)
- 2170. [Morcelle une chaîne](#)
- 2171. [Mot la valeur d'un champs.](#)
- 2172. [Multiplie deux nombres de taille arbitraire.](#)

N

- 2173. [Nom de l'utilisateur actuellement identifié.](#)
- 2174. [Nom de la base de données.](#)
- 2175. [Nombre de colonnes dans un résultat](#)
- 2176. [Nombre de ligne dans un résultat.](#)
- 2177. [Nomme le frame courant.](#)

O

- 2178. [Object id de la racine.](#)
- 2179. [Object record de hw document.](#)
- 2180. [object records d'un groupe d'enfants.](#)
- 2181. [Ouvre un nouveau document PDF.](#)
- 2182. [Optimise une base.](#)
- 2183. [Options disponibles pour les expressions régulières.](#)
- 2184. [Ouverture d'un fichier ou d'une URL.](#)
- 2185. [Ouverture d'une base dBase.](#)
- 2186. [Ouvre la connexion à l'historique système.](#)
- 2187. [Ouvre le module de l'algorithme et le mode à utiliser](#)
- 2188. [Ouvre un document FDF.](#)
- 2189. [Ouvre un dossier, et récupère un pointeur dessus.](#)
- 2190. [Ouvre un fichier compressé](#)
- 2191. [Ouvre un flot IMAP vers une boîte aux lettres.](#)
- 2192. [Ouvre un flot IMAP vers une nouvelle boîte aux lettres.](#)
- 2193. [Ouvre un nouveau document PDF.](#)
- 2194. [Ouvre un nouveau fichier Shockwave Flash](#)
- 2195. [Ouvre un objet de grande taille.](#)
- 2196. [Ouvre un objet SLOB.](#)
- 2197. [Ouvre un pointeur Oracle.](#)
- 2198. [Ouvre une base de données dbm](#)

- 2199. [Ouvre une base de données.](#)
- 2200. [Ouvre une connexion à un serveur Informix.](#)
- 2201. [Ouvre une connexion à un serveur MS SQL server.](#)
- 2202. [Ouvre une connexion à un serveur MySQL.](#)
- 2203. [Ouvre une connexion à un serveur Sybase.](#)
- 2204. [Ouvre une connexion à une base de données Interbase.](#)
- 2205. [Ouvre une connexion FTP](#)
- 2206. [Ouvre une connexion Hyperwave.](#)
- 2207. [Ouvre une connexion ICAP.](#)
- 2208. [Ouvre une connexion MCAL.](#)
- 2209. [Ouvre une connexion mSQL.](#)
- 2210. [Ouvre une connexion Oracle.](#)
- 2211. [Ouvre une connexion persistante à un serveur Informix.](#)
- 2212. [Ouvre une connexion persistante à un serveur MS SQL.](#)
- 2213. [Ouvre une connexion persistante à un serveur mSQL.](#)
- 2214. [Ouvre une connexion persistante à un serveur MySQL.](#)
- 2215. [Ouvre une connexion persistante à un serveur Sybase.](#)
- 2216. [Ouvre une connexion persistante à une base de données Interbase.](#)
- 2217. [Ouvre une connexion persistante à une base de données.](#)
- 2218. [Ouvre une connexion persistante à une source de données.](#)
- 2219. [Ouvre une connexion.](#)
- 2220. [Ouvre une image créée par les fonctions images PHP.](#)
- 2221. [Ouvre une image GIF.](#)
- 2222. [Ouvre une image JPEG.](#), [Ouvre une image JPEG.](#)
- 2223. [Ouvre une socket de connexion Internet ou Unix persistante.](#)
- 2224. [Ouvre une socket de connexion Internet ou Unix.](#)

P

- 2225. [Perform a CURL session](#)
- 2226. [Place le pointeur de résultat à un offset donné](#)
- 2227. [Place un objet sur la scène.](#)
- 2228. [Place une image dans la page.](#)
- 2229. [Place une image enregistrée dans la page.](#)
- 2230. [Places une image dans la page.](#)
- 2231. [Plus grande valeur aléatoire possible.](#)
- 2232. [Positionne le pointeur de tableau en fin de tableau](#)
- 2233. [Positionne un flag sur un message.](#)
- 2234. [Prépare et exécute une requête SQL.](#)
- 2235. [Prépare une chaîne pour une recherche par expression régulière insensible à la casse.](#)
- 2236. [Prépare une commande pour l'exécution](#)
- 2237. [Prépare une expression régulière pour effectuer une recherche insensible à la casse.](#)
- 2238. [Prépare une requête pour lier les paramètres et l'exécuter ultérieurement.](#)
- 2239. [Prépare une requête SQL](#)
- 2240. [Prépare une requête SQL pour l'exécution.](#)
- 2241. [Prend une ou plusieurs valeurs, au hasard dans un tableau](#)
- 2242. [Prepares for a search](#)
- 2243. [Process a raw transaction with Payflow Pro](#)
- 2244. [Process a transaction with Payflow Pro](#)
- 2245. [Puissance](#)

R

- 2246. [Récupère tous les entêtes des requêtes HTTP.](#)
- 2247. [Répète une chaîne.](#)
- 2248. [Réserve un sémaphore.](#)
- 2249. [Résolution DNS d'une adresse IP.](#)
- 2250. [Racine carrée.](#)
- 2251. [Rassemble plusieurs tableaux](#)
- 2252. [Reçoit tous les objets SNMP d'un agent.](#)
- 2253. [Reçoit un objet SNMP.](#)
- 2254. [Read data from shared memory block](#)
- 2255. [Reads a line](#)
- 2256. [Reads the history](#)
- 2257. [Recherche dans tout l'arbre LDAP.](#)
- 2258. [Recherche dans un seul niveau.](#)
- 2259. [Recherche la dernière occurrence d'un caractère dans une chaîne](#)
- 2260. [Recherche la dernière occurrence d'un caractère dans une chaîne.](#), [Recherche la dernière occurrence d'un caractère dans une chaîne.](#)
- 2261. [Recherche la longueur du premier segment de chaîne qui ne corresponde pas au masque donné.](#)
- 2262. [Recherche la première occurrence d'un caractère.](#)
- 2263. [Recherche par expression régulière insensible à la casse.](#)
- 2264. [Recherche un événement dans le calendrier.](#), [Recherche un événement dans le calendrier.](#)
- 2265. [Recherche un message dans le domaine courant.](#)
- 2266. [Recherche un objet dans un groupe.](#), [Recherche un objet dans un groupe.](#)
- 2267. [Recherche un objet.](#), [Recherche un objet.](#)
- 2268. [Rechercher et remplacer par expression régulière standard.](#)
- 2269. [Recode de fichier à fichier, en fonction de la requête.](#)
- 2270. [Recode une chaîne en fonction de la requête.](#)
- 2271. [Recule d'un frame.](#)
- 2272. [Recule le pointeur courant de tableau](#)
- 2273. [Registers a completion function](#)
- 2274. [Regroupe tous les éléments d'un tableau dans une chaîne, avec une chaîne de jointure.](#), [Regroupe tous les éléments d'un tableau dans une chaîne, avec une chaîne de jointure.](#)
- 2275. [Rel'che un segment de mémoire partagée.](#)
- 2276. [Remet le pointeur interne de tableau au début](#)
- 2277. [Remplace dans une sous partie de chaîne](#)
- 2278. [Remplace le domaine courant.](#)
- 2279. [Remplace le domaine lors d'une recherche.](#)
- 2280. [Remplace les clés par les valeurs, et les valeurs par les clés](#)
- 2281. [Remplace ou insère une entrée.](#)
- 2282. [Remplace toutes les occurrences d'un caractère par un autre.](#)
- 2283. [Remplace toutes les occurrences d'une chaîne par une autre.](#)
- 2284. [Remplace un enregistrement dans une base dBase.](#)
- 2285. [Remplace une valeur des attributs courants par une autre.](#)
- 2286. [Remplace une valeur.](#)
- 2287. [Remplacement par expression régulière insensible à la casse.](#)
- 2288. [Remplacement par expression régulière.](#)
- 2289. [remplir avec une région avec une couleur spécifique.](#)
- 2290. [Remplis et dessine le chemin courant.](#)
- 2291. [Remplis le chemin courant.](#), [Remplis le chemin courant.](#)

- 2292. [Remplis le chemin, dessine le bord et ferme le chemin.](#)
- 2293. [Remplis le chemin, et dessine le bord.](#)
- 2294. [Remplis, dessine et ferme le chemin courant.](#)
- 2295. [Remplit.](#)
- 2296. [Renomme un fichier sur un serveur FTP.](#)
- 2297. [Renomme un fichier.](#)
- 2298. [Renomme une boîte aux lettres.](#)
- 2299. [Renvoie l'espace disque disponible dans le répertoire.](#)
- 2300. [Renvoie l'heure à laquelle l'inode a été accédé pour la dernière fois.](#)
- 2301. [Renvoie la date à laquelle le fichier a été accédé pour la dernière fois.](#)
- 2302. [Renvoie la date de dernière modification du fichier.](#)
- 2303. [Renvoie la ligne courant sur laquelle se trouve le pointeur du fichier et élimine les balises HTML.](#)
- 2304. [Renvoie la ligne courante sur laquelle se trouve le pointeur du fichier et cherche dans le résultat les champs CSV](#)
- 2305. [Renvoie la ligne courante sur laquelle se trouve le pointeur du fichier.](#)
- 2306. [Renvoie la position du pointeur du fichier.](#)
- 2307. [Renvoie la racine carrée d'un nombre de taille arbitraire.](#)
- 2308. [Renvoie la taille du fichier.](#)
- 2309. [Renvoie le caractère que pointe le pointeur du fichier.](#)
- 2310. [Renvoie le nom du dossier.](#)
- 2311. [Renvoie le nom du fichier vers lequel pointe un lien symbolique.](#)
- 2312. [Renvoie le nom du possesseur du fichier.](#)
- 2313. [Renvoie le numéro d'inode du fichier.](#)
- 2314. [Renvoie les informations à propos d'un fichier ou d'un lien symbolique.](#)
- 2315. [Renvoie les informations à propos d'un fichier.](#)
- 2316. [Renvoie les informations à propos d'un lien.](#)
- 2317. [Renvoie les permissions affectées au fichier.](#)
- 2318. [Replace le pointeur courant au début du fichier](#)
- 2319. [Replace le pointeur de fichier au début.](#)
- 2320. [Représente un id globale en un id virtuel local.](#)
- 2321. [Restaure un environnement sauvé.](#)
- 2322. [Restaures un environnement.](#)
- 2323. [Restores the previous error handler function](#)
- 2324. [Retarde l'exécution en micro-secondes](#)
- 2325. [Retarde l'exécution.](#)
- 2326. [Retourne à la première entrée du dossier.](#)
- 2327. [Retourne chaque paire clé/valeur d'un tableau](#)
- 2328. [Retourne des informations sur les caractères utilisés dans une chaîne.](#)
- 2329. [Retourne des informations sur un groupe., Retourne des informations sur un groupe.](#)
- 2330. [Retourne des informations sur un utilisateur., Retourne des informations sur un utilisateur.](#)
- 2331. [Retourne l' UID d'un message.](#)
- 2332. [Retourne l'élément courant d'un tableau](#)
- 2333. [Retourne l'adresse IP correspondant à un hôte.](#)
- 2334. [Retourne l'entête d'un message.](#)
- 2335. [Retourne l'heure actuelle](#)
- 2336. [Retourne l'ID de l'utilisateur du processus courant.](#)
- 2337. [Retourne l'id du groupe de processus.](#)
- 2338. [Retourne l'ID effectif du groupe du processus courant.](#)
- 2339. [Retourne l'identifiant du groupe de processus.](#)
- 2340. [Retourne l'identifiant du premier attribut.](#)
- 2341. [Retourne l'identifiant du processus courant.](#)

- 2342. [Retourne l'identifiant du processus parent.](#)
- 2343. [Retourne l'identifiant généré par la dernière requête INSERT.](#)
- 2344. [retourne l'identifiant maximal de hash.](#)
- 2345. [Retourne l'index de l'octet courant d'un analyseur XML.](#)
- 2346. [Retourne l'index de la couleur d'un pixel donné.](#)
- 2347. [Retourne l'index de la couleur donnée, ou la plus proche possible. .](#)
- 2348. [Retourne l'index de la couleur donnée.](#)
- 2349. [Retourne l'index de la couleur la plus proche d'une couleur donnée.](#)
- 2350. [retourne l'inode du script.](#)
- 2351. [Retourne l'UID du groupe du processus courant.](#)
- 2352. [retourne l'UID du propriétaire du script actuel.](#)
- 2353. [Retourne l'UID effectif de l'utilisateur du processus courant.](#)
- 2354. [Retourne l'URL d'une animation Shockwave Flash.](#)
- 2355. [Retourne la configuration actuelle de l'option magic_quotes_runtime.](#)
- 2356. [Retourne la couleur associée à un index.](#)
- 2357. [Retourne la date de dernière modification d'un fichier sur un serveur FTP.](#)
- 2358. [retourne la date de dernière modification de la page.](#)
- 2359. [Retourne la date/heure](#)
- 2360. [Retourne la dernière erreur \(si elle existe\) qui est survenu lors de la dernière requête.](#)
- 2361. [Retourne la dernière erreur de stmt|conn|global.](#)
- 2362. [Retourne la hauteur de l'image.](#)
- 2363. [Retourne la hauteur de la police.](#)
- 2364. [Retourne la hauteur du A majuscule, et du x minuscule.](#)
- 2365. [Retourne la largeur d'une image.](#)
- 2366. [Retourne la largeur de la police.](#)
- 2367. [Retourne la largeur du texte avec la police courante.](#)
- 2368. [Retourne la ligne associée.](#)
- 2369. [Retourne la ligne suivante dans un tableau.](#)
- 2370. [Retourne la liste d'IP correspondants à un hôte.](#)
- 2371. [Retourne la liste des arguments d'une fonction](#)
- 2372. [Retourne la liste des fichiers dans un dossier.](#)
- 2373. [Retourne la longueur d'un champs.](#)
- 2374. [Retourne la longueur d'une chaîne.](#)
- 2375. [Retourne la longueur de la chaîne.](#)
- 2376. [Retourne la longueur du champs spécifié.](#)
- 2377. [Retourne la longueur du premier segment qui vérifie le masque.](#)
- 2378. [Retourne la position courante du pointeur interne](#)
- 2379. [Retourne la prochaine occurrence d'une événement.](#)
- 2380. [Retourne la structure d'info par défaut d'un document PDF.](#)
- 2381. [Retourne la table de traduction utilisée par @ref{function.htmlspecialchars . . htmlspecialchars\(\)} et @ref{function.htmlentities . . htmlentities\(\)}.](#)
- 2382. [Retourne la taille d'un champs.](#)
- 2383. [Retourne la taille d'un fichier.](#)
- 2384. [Retourne la taille d'une colonne](#)
- 2385. [Retourne la taille d'une image GIF, JPG ou PNG.](#)
- 2386. [Retourne la taille de bloc d'un algorithme, Retourne la taille de bloc d'un algorithme](#)
- 2387. [Retourne la taille de bloc d'un chiffrement.](#)
- 2388. [Retourne la taille de bloc du hash.](#)
- 2389. [Retourne la taille de chaque colonne d'une ligne de résultat.](#)
- 2390. [Retourne la taille de la chaîne.](#)
- 2391. [Retourne la taille de la clé d'un chiffrement.](#)

- 2392. [Retourne la taille de la colonne.](#)
- 2393. [Retourne la taille du VI d'un algorithme](#)
- 2394. [Retourne la taille du VI utilisé par un couple chiffrement/mode](#)
- 2395. [Retourne la taille imprimée.](#)
- 2396. [Retourne la taille interne de stockage d'un champs donné.](#)
- 2397. [Retourne la taille maximale de clé](#)
- 2398. [Retourne la taille maximale de la clé pour un mode](#)
- 2399. [Retourne la valeur ASCII du caractère.](#)
- 2400. [Retourne la valeur d'un champs.](#)
- 2401. [Retourne la valeur d'une colonne dans une ligne lue](#)
- 2402. [Retourne la valeur d'une option de PHP](#)
- 2403. [retourne la valeur de la variable d'environnement.](#)
- 2404. [Retourne la valeur de la variable, au format chaîne.](#)
- 2405. [Retourne la valeur de pi](#)
- 2406. [Retourne la valeur numérique \(double\) de la variable.](#)
- 2407. [Retourne la valeur numérique \(integer\) de la variable.](#)
- 2408. [Retourne le chemin canonique absolu](#)
- 2409. [Retourne le chemin du terminal.](#)
- 2410. [Retourne le code d'erreur de la dernière requête Informix.](#)
- 2411. [Retourne le code d'erreur Oracle.](#)
- 2412. [Retourne le code d'erreur.](#)
- 2413. [retourne le configuration actuelle de l'option magic_quotes_gpc.](#)
- 2414. [Retourne le contenu d'un objet BLOB.](#)
- 2415. [Retourne le contenu d'un objet char.](#)
- 2416. [Retourne le contenu de la variable sqlca.sqlerrd\[0..5\] après une requête.](#)
- 2417. [Retourne le couple \(clé ; valeur\) suivant d'une carte donnée.](#)
- 2418. [Retourne le dernier identifiant d'objet.](#)
- 2419. [Retourne le dernier message d'erreur du serveur \(min_message_severity?\).](#)
- 2420. [Retourne le dernier message du serveur](#)
- 2421. [Retourne le domaine NIS par défaut.](#)
- 2422. [Retourne le fichier courant, ou la position courante.](#)
- 2423. [Retourne le flag d'un champs.](#)
- 2424. [Retourne le logo](#)
- 2425. [Retourne le logo de Zend](#)
- 2426. [Retourne le message d'erreur](#)
- 2427. [Retourne le message d'erreur de la dernière requête Informix.](#)
- 2428. [Retourne le message d'erreur Oracle.](#)
- 2429. [Retourne le message LDAP de la dernière commande LDAP.](#)
- 2430. [retourne le niveau d'utilisation des ressources.](#)
- 2431. [Retourne le nom d'hôte correspondant à une IP.](#)
- 2432. [Retourne le nom d'hôte.](#)
- 2433. [Retourne le nom d'un champs., Retourne le nom d'un champs.](#)
- 2434. [Retourne le nom d'une colonne, Retourne le nom d'une colonne](#)
- 2435. [Retourne le nom d'une colonne.](#)
- 2436. [Retourne le nom d'une table à partir d'un nom de champs., Retourne le nom d'une table à partir d'un nom de champs.](#)
- 2437. [Retourne le nom de device du terminal.](#)
- 2438. [Retourne le nom de l'algorithme](#)
- 2439. [Retourne le nom de la colonne de résultat.](#)
- 2440. [Retourne le nom de la machine maître pour une carte.](#)
- 2441. [Retourne le nom de la table où se trouve une colonne](#)

- 2442. [Retourne le nom de la table qui contient le champs spécifié.](#)
- 2443. [Retourne le nom de login.](#)
- 2444. [Retourne le nom de protocole associé au numéro de protocole](#)
- 2445. [Retourne le nom de tty.](#)
- 2446. [Retourne le nom du curseur](#)
- 2447. [Retourne le nom du dossier courant.](#)
- 2448. [Retourne le nom du hash.](#)
- 2449. [Retourne le nom du mode](#)
- 2450. [Retourne le nom du mois.](#)
- 2451. [Retourne le nom du possesseur du script courant.](#)
- 2452. [Retourne le nom du système.](#)
- 2453. [Retourne le nombre courant de colonne d'un analyseur XML.](#)
- 2454. [Retourne le nombre courant de colonne d'un analyseur XML. .](#)
- 2455. [Retourne le nombre d'élément d'un tableau](#)
- 2456. [Retourne le nombre d'arguments passés à une fonction.](#)
- 2457. [Retourne le nombre de champs](#)
- 2458. [Retourne le nombre de champs d'un résultat.](#)
- 2459. [Retourne le nombre de champs dans un résultat.,](#) [Retourne le nombre de champs dans un résultat.,](#)
[Retourne le nombre de champs dans un résultat.,](#) [Retourne le nombre de champs dans un résultat.](#)
- 2460. [Retourne le nombre de champs dans une base filePro.,](#) [Retourne le nombre de champs dans une base filePro.](#)
- 2461. [Retourne le nombre de colonnes](#)
- 2462. [Retourne le nombre de colonnes dans un résultat](#)
- 2463. [Retourne le nombre de colonnes dans une requête.](#)
- 2464. [Retourne le nombre de jour d'un mois.](#)
- 2465. [Retourne le nombre de jour entre le 21 Mars et P'ques. pour une année donnée.](#)
- 2466. [Retourne le nombre de ligne affectées.](#)
- 2467. [Retourne le nombre de ligne d'un résultat.](#)
- 2468. [Retourne le nombre de lignes affectées lors de la dernière opération SQL.](#)
- 2469. [Retourne le nombre de lignes affectées par la dernière requête.](#)
- 2470. [Retourne le nombre de lignes affectées par une modification](#)
- 2471. [Retourne le nombre de lignes affectées par une requête.](#)
- 2472. [Retourne le nombre de lignes affectées.](#)
- 2473. [Retourne le nombre de lignes dans un résultat.,](#) [Retourne le nombre de lignes dans un résultat.,](#)
[Retourne le nombre de lignes dans un résultat.,](#) [Retourne le nombre de lignes dans un résultat.,](#)
[Retourne le nombre de lignes dans un résultat.](#)
- 2474. [Retourne le nombre de lignes.](#)
- 2475. [Retourne le nombre de message dans la boîte aux lettres courante.](#)
- 2476. [Retourne le nombre de messages récents dans la boîte aux lettres courante.](#)
- 2477. [Retourne le nombre de tuples affectés.](#)
- 2478. [Retourne le numéro d'erreur LDAP de la dernière commande exécutée.](#)
- 2479. [Retourne le numéro d'ordre d'une carte.](#)
- 2480. [Retourne le numéro d'une colonne.](#)
- 2481. [Retourne le numéro de colonne](#)
- 2482. [Retourne le numéro de frame courant.](#)
- 2483. [retourne le numéro de la version courante de PHP.](#)
- 2484. [Retourne le numéro de ligne courant d'un analyseur XML.](#)
- 2485. [Retourne le numéro de message d'erreur de la dernière opération MySQL.](#)
- 2486. [Retourne le numéro de port associé à un service Internet. et un protocole.](#)
- 2487. [Retourne le numéro de port.](#)
- 2488. [retourne le numéro de processus courant.](#)

- 2489. [Retourne le numéro de protocole associé au nom de protocole](#)
- 2490. [Retourne le numéro de séquence de message pour un UID donné.](#)
- 2491. [Retourne le numéro du jour de la semaine.](#)
- 2492. [Retourne le premier attribut.](#)
- 2493. [Retourne le premier couple \(clé ; valeur\) d'une carte donnée.](#)
- 2494. [Retourne le prochain identifiant d'objet libre.](#)
- 2495. [Retourne le rectangle entourant un texte et dessiné avec une police PostScript Type1.](#)
- 2496. [retourne le rectangle entourant un texte et dessiné avec une police TrueType.](#)
- 2497. [Retourne le reste d'une division entre nombre de taille arbitraire.](#)
- 2498. [Retourne le sémaphore associé à la colonne spécifiée dans le résultat courant.](#)
- 2499. [Retourne le service Internet qui correspond au port et protocole.](#)
- 2500. [Retourne le sid du processus.](#)
- 2501. [Retourne le texte associée avec l'erreur générée lors de la dernière requête.](#)
- 2502. [Retourne le timestamp UNIX actuel avec microsecondes.](#)
- 2503. [Retourne le timestamp UNIX actuel.](#)
- 2504. [Retourne le timestamp UNIX d'une date GMT.](#)
- 2505. [Retourne le timestamp UNIX d'une date.](#)
- 2506. [Retourne le type d'un champs donné par index.](#)
- 2507. [Retourne le type d'un champs.](#)
- 2508. [Retourne le type de champs.](#)
- 2509. [Retourne le type de commande OCI.](#)
- 2510. [Retourne le type de données d'une colonne.](#)
- 2511. [Retourne le type de fichier](#)
- 2512. [Retourne le type de la colonne de résultat.](#)
- 2513. [Retourne le type de la colonne spécifiée dans le résultat courant.](#)
- 2514. [Retourne le type de la variable.](#)
- 2515. [Retourne le type numérique d'une colonne](#)
- 2516. [Retourne les ancrages qui pointent sur un objet.](#)
- 2517. [Retourne les attributs d'une entrée d'un résultat.](#)
- 2518. [Retourne les attributs, et verrouille l'objet.](#)
- 2519. [Retourne les bits de status de la connexion.](#)
- 2520. [Retourne les données de résultat.](#)
- 2521. [Retourne les données enregistrées dans une colonne, à partir d'un résultat, et retourne un objet.](#)
- 2522. [Retourne les enregistrements MX d'un hôte.](#)
- 2523. [Retourne les entêtes de tous les messages d'une boîte aux lettres.](#)
- 2524. [Retourne les fils d'un document distant.](#)
- 2525. [Retourne les identifiants du groupe du processus courant.](#)
- 2526. [Retourne les informations de statut sur une boîte aux lettres autre que la boîte courante.](#)
- 2527. [Retourne les informations sur une socket](#)
- 2528. [Retourne les lignes résultats sous la forme d'un objet.](#)
- 2529. [Retourne les limites système.](#)
- 2530. [Retourne les noms des méthodes d'une classe.](#)
- 2531. [Retourne les options.](#)
- 2532. [Retourne les types d'images supportés par la version courante de PHP](#)
- 2533. [Retourne les valeurs d'un identifiant de résultat.](#)
- 2534. [Retourne les valeurs d'un tableau](#)
- 2535. [Retourne les valeurs des attributs d'une classe.](#)
- 2536. [Retourne les valeurs par défaut des attributs d'une classe.](#)
- 2537. [Retourne toutes les alertes \(si elles existent\) qui sont survenues lors de la dernière requête, ou depuis que la pile d'alerte a été réinitialisée.](#)
- 2538. [Retourne toutes les clés d'un tableau](#)

- 2539. [Retourne toutes les entrées d'un résultat.](#)
- 2540. [Retourne toutes les entrées.](#)
- 2541. [Retourne toutes les erreurs \(si elles existent\) qui sont survenus lors de la dernière requête, ou depuis que la pile d'erreur a été réinitialisée.](#)
- 2542. [Retourne toutes les lignes d'un résultat.](#)
- 2543. [Retourne toutes les valeurs binaires à partir d'un identifiant de résultat.](#)
- 2544. [Retourne TRUE si la fonction a été définie.](#)
- 2545. [Retourne TRUE si le client a abandonné la connexion.](#)
- 2546. [Retourne TRUE si le script a expiré.](#)
- 2547. [Retourne un caractère.](#)
- 2548. [Retourne un champs d'un résultat.](#)
- 2549. [Retourne un document distant.](#)
- 2550. [Retourne un document texte.,](#) [Retourne un document texte.](#)
- 2551. [Retourne un document.](#)
- 2552. [Retourne un identifiant de sémaphore.](#)
- 2553. [Retourne un identifiant de type de serveur FTP.](#)
- 2554. [Retourne un message d'erreur.](#)
- 2555. [Retourne un ND d'une entrée d'un résultat.](#)
- 2556. [Retourne un nouveau pointeur à utiliser pour lier les pointeurs de références \(ref-cursors\)](#)
- 2557. [Retourne un objet contenant la structure de date pour le flot courant.](#)
- 2558. [Retourne un tableau avec les résultat de la recherche.](#)
- 2559. [Retourne un tableau contenant les tailles de clés acceptées par un algorithme](#)
- 2560. [Retourne un tableau de message après recherche.](#)
- 2561. [Retourne un tableau dont les éléments sont classés en sens inverse.](#)
- 2562. [Retourne un timestamp UNIX pour P'ques. à minuit, pour une année donnée.](#)
- 2563. [Retourne une adresse email proprement formatée, à partir du nom de la boîte aux lettre, de l'hôte, et des informations personnelles.](#)
- 2564. [Retourne une chaîne contenant les informations de version du serveur.](#)
- 2565. [Retourne une chaîne formatée.](#)
- 2566. [Retourne une clé d'un tableau associatif](#)
- 2567. [Retourne une donnée d'une ligne lue.](#)
- 2568. [Retourne une ligne de résultat sous la forme d'un tableau associatif.](#)
- 2569. [Retourne une ligne de résultat sous la forme d'un tableau.](#)
- 2570. [Retourne une ligne de résultat.](#)
- 2571. [Retourne une ligne sous la forme d'un objet.,](#) [Retourne une ligne sous la forme d'un objet.,](#) [Retourne une ligne sous la forme d'un objet.](#)
- 2572. [Retourne une ligne sous la forme d'un tableau énuméré.,](#) [Retourne une ligne sous la forme d'un tableau énuméré.](#)
- 2573. [Retourne une ligne sous la forme d'un tableau.](#)
- 2574. [Retourne une liste d'événement entre deux dates.,](#) [Retourne une liste d'événement entre deux dates.](#)
- 2575. [Retourne une liste d'événements qui ont une alarme prévue à une date.,](#) [Retourne une liste d'événements qui ont une alarme prévue à une date.](#)
- 2576. [Retourne une partie de la chaîne.](#)
- 2577. [Retourne une section extraite du corps d'un message.](#)
- 2578. [Retourne une tableau contenant la liste des arguments d'une fonction.](#)
- 2579. [Retourne vrai si une valeur appartient à un tableau](#)
- 2580. [Return a string describing a socket error.](#)
- 2581. [Return the contents of the output buffer](#)
- 2582. [Return the current CURL version](#)
- 2583. [Return the length of the output buffer](#)
- 2584. [Returns a positive association ID on success; zero on failure](#)

- 2585. [Returns a record](#)
- 2586. [Returns additional error information](#)
- 2587. [Returns error description](#)
- 2588. [Returns error number](#)
- 2589. [Returns number of hits for last search](#)
- 2590. [Returns the version of the Payflow Pro software](#)
- 2591. [Rotation de la transformation courante.](#)

S

- 2592. [Sélectionne la police et sa taille.](#)
- 2593. [Sélectionne une base de données mSQL., Sélectionne une base de données mSQL.](#)
- 2594. [Sélectionne une base de données MySQL.](#)
- 2595. [Sélectionne une base de données Sybase.](#)
- 2596. [Sélectionne une nouvelle zone pour un dessin ultérieur.](#)
- 2597. [Sépare le nom du fichier et le nom du dossier.](#)
- 2598. [Sauve l'environnement courant.](#)
- 2599. [Sauver un document FDF.](#)
- 2600. [Scinde un ND en plusieurs composants.](#)
- 2601. [Scinde une chaîne en morceau, gr'ce à un délimiteur.](#)
- 2602. [Scinde une chaîne en plus petits morceaux.](#)
- 2603. [Scinde une chaîne en un tableau, gr'ce à une expression régulière.](#)
- 2604. [Se connecte à un serveur LDAP.](#)
- 2605. [Se connecte à un serveur Oracle avec une nouvelle connexion. Retourne une nouvelle session.](#)
- 2606. [Se lie à un serveur LDAP.](#)
- 2607. [See if an exception was caught from the previous function](#)
- 2608. [Selectionne la base de données MS SQL.](#)
- 2609. [Selectionne la police courante, et sa taille.](#)
- 2610. [send mail](#)
- 2611. [Set an option for a CURL transfer](#)
- 2612. [Sets a user-defined error handler function.](#)
- 2613. [Shuts down the Payflow Pro library](#)
- 2614. [Sinus](#)
- 2615. [Souscrit à une boîte aux lettres.](#)
- 2616. [Soustrait un nombre de taille arbitraire à un autre.](#)
- 2617. [Specifies the maximum number of records to retrieve](#)
- 2618. [Specifies the preferred record syntax for retrieval](#)
- 2619. [suggère l'orthographe d'un mot](#)
- 2620. [Suggest spellings of a word](#)
- 2621. [Supprime la récurrence de la structure globale.](#)
- 2622. [Supprime un alias.](#)
- 2623. [Supprime un flag sur un message.](#)
- 2624. [Supprime un objet BLOB.](#)
- 2625. [Supprime un objet char.](#)
- 2626. [Supprime un objet SLOB.](#)
- 2627. [Supprime un segment de mémoire partagée sous Unix.](#)
- 2628. [Supprime un utilisateur virtuel.](#)
- 2629. [Supprime une variable dans la session courante](#)
- 2630. [Synchronise une base de données.](#)
- 2631. [Synonyme de @ref{function.odbc-exec ., odbc_exec\(\)}](#)

T

- 2632. [Télécharge un fichier depuis un serveur FTP et le sauve dans un fichier déjà ouvert.](#)
- 2633. [Télécharge un fichier depuis un serveur FTP.](#)
- 2634. [Taille d'un document.](#)
- 2635. [Tangente](#)
- 2636. [Termine l'action courante.](#)
- 2637. [Termine la définition de symbole.](#)
- 2638. [Termine la définition du bouton courant.](#)
- 2639. [Termine la liaison avec un serveur LDAP.](#)
- 2640. [Termine la souscription à une boîte aux lettres.](#)
- 2641. [Termine le script courant.](#)
- 2642. [Termine un cryptage](#)
- 2643. [Termine un document.](#)
- 2644. [Termine un flot IMAP.](#)
- 2645. [Termine une connexion PostgreSQL.](#)
- 2646. [Termine une page.,](#) [Termine une page.](#)
- 2647. [Termine une section de texte.](#)
- 2648. [Test un module ouvert](#)
- 2649. [Test le cryptage par bloc d'un algorithme](#)
- 2650. [Teste la fin d'un fichier compressé.](#)
- 2651. [Teste la fin du fichier.](#)
- 2652. [Teste le cryptage par bloc d'un mode](#)
- 2653. [Teste si la valeur d'une colonne est NULL](#)
- 2654. [Teste si le mode retourne les données par bloc](#)
- 2655. [Teste si un champs est à NULL.](#)
- 2656. [Teste un mode](#)
- 2657. [Transforme une chaîne ASCII en EBCDIC](#)
- 2658. [Transforme une chaîne EBCDIC en ASCII](#)
- 2659. [Transforme une liste de variables en tableau](#)
- 2660. [Transforme une variable en tableau](#)
- 2661. [Translate la transformation courante.](#)
- 2662. [Tri d'un tableau avec l'algorithme à "ordre naturel"](#)
- 2663. [Tri d'un tableau avec l'algorithme à "ordre naturel" insensible à la casse](#)
- 2664. [Tri multi-dimensionnel](#)
- 2665. [Trie des messages.](#)
- 2666. [Trie en ordre inverse](#)
- 2667. [Trie le tableau](#)
- 2668. [Trie les clés d'un tableau en utilisant une fonction de comparaison définie par l'utilisateur](#)
- 2669. [Trie les valeurs d'un tableau en utilisant une fonction de comparaison définie par l'utilisateur](#)
- 2670. [Trie un tableau en ordre](#)
- 2671. [Trie un tableau en ordre inverse](#)
- 2672. [Trie un tableau en sens inverse et suivant les clés](#)
- 2673. [Trie un tableau en utilisant une fonction de comparaison définie par l'utilisateur.](#)
- 2674. [Trie un tableau suivant les clés](#)
- 2675. [Tronque une fichier](#)
- 2676. [Trouve la première occurrence d'une chaîne.](#)
- 2677. [Turn implicit flush on/off](#)
- 2678. [Turn on output buffering](#)
- 2679. [Type de données d'un champs.](#)

U

- 2680. [Use CORBA enums](#)
- 2681. [Use CORBA structs](#)
- 2682. [Utilisation des ressources.](#)
- 2683. [Utilise un analyseur XML à l'intérieur d'un objet.](#)
- 2684. [Utilise une variable PHP pour la phase de définition, dans un SELECT.](#)
- 2685. [Utilise une variable PHP pour la phase de définition, dans une commande SELECT.](#)

V

- 2686. [Vérifie le courrier de la boîte aux lettres courante.](#)
- 2687. [Vérifie qu'un identifiant d'objet est dans un groupe.](#)
- 2688. [Vérifie qu'une clé existe.](#)
- 2689. [Vérifie qu'une constante existe.](#)
- 2690. [Vérifie que l'année est bissextile.](#)
- 2691. [Vérifie que la méthode existe pour une classe.](#)
- 2692. [Vérifie que le flot IMAP est toujours actif.](#)
- 2693. [Vérifie si un fichier existe.](#)
- 2694. [Vérifie un mot](#)
- 2695. [Vérifie un mot sans en changer la casse et sans essayer de supprimer les espaces aux extrémités.](#)
- 2696. [Valeur absolue](#)
- 2697. [Valide les transactions en cours.](#)
- 2698. [Valide une date.](#)
- 2699. [Valide une date/heure.](#)
- 2700. [Valide une heure.](#)
- 2701. [Valide une transaction](#)
- 2702. [Valide une transaction ODBC](#)
- 2703. [Valide une transaction Oracle.](#)
- 2704. [Verrouille le fichier.](#)
- 2705. [Vide les buffers de sorties.](#)

W

- 2706. [Write data into shared memory block](#)
- 2707. [Writes the history](#)

- 2742. [Classer un tableau multidimensionnel](#)
- 2743. [Closing shared memory block](#)
- 2744. [Compactage d'une chaîne](#)
- 2745. [Compter le nombre de hit d'un utilisateur.](#)
- 2746. [Connaître le titre du page distante](#)
- 2747. [Connexion à un serveur Informix](#)
- 2748. [Connexion à un serveur Ovrimos SQL Server et préparation d'une requête](#)
- 2749. [Création d'un document PDF avec pdflib 2.x](#)
- 2750. [Création d'une base dBase](#)
- 2751. [Create a new shared memory block](#)
- 2752. [Creation d'un document PDF avec pdflib 0.6](#)
- 2753. [Crypte une valeur avec un TripleDES, en mode ECB.](#)

D

- 2754. [Définition d'une constante](#)
- 2755. [Définition de constantes](#)
- 2756. [dbm](#)
- 2757. [Deleting shared memory block](#)
- 2758. [Deuxièmes nouvelles balises start/end](#)

E

- 2759. [Eclatement d'une chaîne de recherche.](#)
- 2760. [Effacer une ressource existante](#)
- 2761. [Encryption d'une valeur avec TripleDES sous 2.4.x en mode ECB](#)
- 2762. [Enregistrer une valeur simple](#)
- 2763. [Entité externe](#)
- 2764. [Enumérer tous les messages d'erreur LDAP](#)
- 2765. [Envoi de eMail avec des entêtes supplémentaires.](#)
- 2766. [Envoi de mail complexe.](#)
- 2767. [Envoi de mail.](#)
- 2768. [Evaluer un document FDF](#)
- 2769. [exemple de la fonction phpversion\(\)](#)
- 2770. [Exemple](#)
- 2771. [Exemple @ref{function.easter-date , , easter_date\(\) }](#)
- 2772. [Exemple @ref{function.fscanf , , fscanf\(\) }](#)
- 2773. [Exemple @ref{function.ip2long , , ip2long\(\) }](#)
- 2774. [Exemple @ref{function.mktime , , mktime\(\) }](#)
- 2775. [Exemple @ref{function.mssql-tablename , , mssql_tablename\(\) }](#)
- 2776. [Exemple @ref{function.mysql-tablename , , mysql_tablename\(\) }](#)
- 2777. [Exemple @ref{function.realpath , , realpath\(\) }](#)
- 2778. [Exemple @ref{function.socket-set-timeout , , socket_set_timeout\(\) }](#)
- 2779. [Exemple @ref{function.strftime , , strftime\(\) }](#)
- 2780. [Exemple avec @ref{function.addcslashes , , addcslashes\(\) }](#)
- 2781. [Exemple avec @ref{function.array-count-values , , array_count_values\(\) }](#)
- 2782. [Exemple avec @ref{function.array-diff , , array_diff\(\) }](#)
- 2783. [Exemple avec @ref{function.array-flip , , array_flip\(\) }](#)
- 2784. [Exemple avec @ref{function.array-intersect , , array_intersect\(\) }](#)
- 2785. [Exemple avec @ref{function.array-keys , , array_keys\(\) }](#)

- 2786. [Exemple avec @ref{function.array-merge .. array_merge\(\)}](#)
- 2787. [Exemple avec @ref{function.array-merge-recursive .. array_merge_recursive\(\)}](#)
- 2788. [Exemple avec @ref{function.array-pad .. array_pad\(\)}](#)
- 2789. [Exemple avec @ref{function.array-pop .. array_pop\(\)}](#)
- 2790. [Exemple avec @ref{function.array-push .. array_push\(\)}](#)
- 2791. [Exemple avec @ref{function.array-rand .. array_rand\(\)}](#)
- 2792. [Exemple avec @ref{function.array-reverse .. array_reverse\(\)}](#)
- 2793. [Exemple avec @ref{function.array-shift .. array_shift\(\)}](#)
- 2794. [Exemple avec @ref{function.array-slice .. array_slice\(\)}](#)
- 2795. [Exemple avec @ref{function.array-unique .. array_unique\(\)}](#)
- 2796. [Exemple avec @ref{function.array-walk .. array_walk\(\)}](#)
- 2797. [Exemple avec @ref{function.arsort .. arsort\(\)}](#)
- 2798. [Exemple avec @ref{function.asort .. asort\(\)}](#)
- 2799. [Exemple avec @ref{function.basename .. basename\(\)}](#)
- 2800. [Exemple avec @ref{function.chop .. chop\(\)}](#)
- 2801. [Exemple avec @ref{function.chr .. chr\(\)}](#)
- 2802. [Exemple avec @ref{function.chunk-split .. chunk_split\(\)}](#)
- 2803. [Exemple avec @ref{function.compact .. compact\(\)}](#)
- 2804. [Exemple avec @ref{function.copy .. copy\(\)}](#)
- 2805. [Exemple avec @ref{function.count .. count\(\)}](#)
- 2806. [Exemple avec @ref{function.date .. date\(\)}](#)
- 2807. [Exemple avec @ref{function.die .. die\(\)}](#)
- 2808. [Exemple avec @ref{function.dirname .. dirname\(\)}](#)
- 2809. [Exemple avec @ref{function.diskfreespace .. diskfreespace\(\)}](#)
- 2810. [Exemple avec @ref{function.echo .. echo\(\)}](#)
- 2811. [Exemple avec @ref{function.ereg-replace .. ereg_replace\(\)}](#)
- 2812. [Exemple avec @ref{function.error-log .. error_log\(\)}](#)
- 2813. [Exemple avec @ref{function.explode .. explode\(\)}](#)
- 2814. [Exemple avec @ref{function.get-browser .. get_browser\(\)}](#)
- 2815. [Exemple avec @ref{function.gmtime .. gmtime\(\)}](#)
- 2816. [Exemple avec @ref{function.gmstrftime .. gmstrftime\(\)}](#)
- 2817. [Exemple avec @ref{function.ibase-connect .. ibase_connect\(\)}](#)
- 2818. [Exemple avec @ref{function.implode .. implode\(\)}](#)
- 2819. [Exemple avec @ref{function.in-array .. in_array\(\)}](#)
- 2820. [Exemple avec @ref{function.krsort .. krsort\(\)}](#)
- 2821. [Exemple avec @ref{function.ksort .. ksort\(\)}](#)
- 2822. [Exemple avec @ref{function.list .. list\(\)}](#)
- 2823. [Exemple avec @ref{function.mcrypt-encrypt .. mcrypt_encrypt\(\)}](#)
- 2824. [Exemple avec @ref{function.mcrypt-list-algorithms .. mcrypt_list_algorithms\(\)}](#)
- 2825. [Exemple avec @ref{function.mcrypt-list-modes .. mcrypt_list_modes\(\)}](#)
- 2826. [Exemple avec @ref{function.mcrypt-module-open .. mcrypt_module_open\(\)}](#)
- 2827. [Exemple avec @ref{function.mdecrypt-generic .. mdecrypt_generic\(\)}](#)
- 2828. [Exemple avec @ref{function.natsort .. natsort\(\)}](#)
- 2829. [Exemple avec @ref{function.ord .. ord\(\)}](#)
- 2830. [Exemple avec @ref{function.ovrimos-connect .. ovrinos_connect\(\)}](#)
- 2831. [Exemple avec @ref{function.ovrimos-result-all .. ovrinos_result_all\(\)}](#)
- 2832. [Exemple avec @ref{function.preg-grep .. preg_grep\(\)}](#)
- 2833. [Exemple avec @ref{function.rawurlencode .. rawurlencode\(\)} example 1](#)
- 2834. [Exemple avec @ref{function.rawurlencode .. rawurlencode\(\)} example 2](#)
- 2835. [Exemple avec @ref{function.rsort .. rsort\(\)}](#)
- 2836. [Exemple avec @ref{function.serialize .. serialize\(\)}](#)

- 2837. [Exemple avec @ref{function.session-name . . session_name\(\) }](#)
- 2838. [Exemple avec @ref{function.set-file-buffer . . set_file_buffer\(\) }](#)
- 2839. [Exemple avec @ref{function.shuffle . . shuffle\(\) }](#)
- 2840. [Exemple avec @ref{function.sort . . sort\(\) }](#)
- 2841. [Exemple avec @ref{function.split . . split\(\) }](#), [Exemple avec @ref{function.split . . split\(\) }](#)
- 2842. [Exemple avec @ref{function.sprintf . . sprintf\(\) }: complété avec des zéros](#)
- 2843. [Exemple avec @ref{function.sprintf . . sprintf\(\) }: format monétaire](#)
- 2844. [Exemple avec @ref{function.sql-regcase . . sql_regcase\(\) }](#)
- 2845. [Exemple avec @ref{function.str-repeat . . str_repeat\(\) }](#)
- 2846. [Exemple avec @ref{function.str-replace . . str_replace\(\) }](#)
- 2847. [Exemple avec @ref{function.strcasecmp . . strcasecmp\(\) }](#)
- 2848. [Exemple avec @ref{function.strchr . . strchr\(\) }](#)
- 2849. [Exemple avec @ref{function.strstr . . strstr\(\) }](#)
- 2850. [Exemple avec @ref{function.strtok . . strtok\(\) }](#)
- 2851. [Exemple avec @ref{function.strtolower . . strtolower\(\) }](#)
- 2852. [Exemple avec @ref{function.strtoupper . . strtoupper\(\) }](#)
- 2853. [Exemple avec @ref{function.strtr . . strtr\(\) }](#)
- 2854. [Exemple avec @ref{function.substr-replace . . substr_replace\(\) }](#)
- 2855. [Exemple avec @ref{function.tempnam . . tempnam\(\) }](#)
- 2856. [Exemple avec @ref{function.touch . . touch\(\) }](#)
- 2857. [Exemple avec @ref{function.ucfirst . . ucfirst\(\) }](#)
- 2858. [Exemple avec @ref{function.ucwords . . ucwords\(\) }](#)
- 2859. [Exemple avec @ref{function.uksort . . uksort\(\) }](#)
- 2860. [Exemple avec @ref{function.unpack . . unpack\(\) }](#)
- 2861. [Exemple avec @ref{function.unserialize . . unserialize\(\) }](#)
- 2862. [Exemple avec @ref{function.urldecode . . urldecode\(\) }](#)
- 2863. [Exemple avec @ref{function.urlencode . . urlencode\(\) }](#)
- 2864. [Exemple avec @ref{function.usort . . usort\(\) }](#)
- 2865. [Exemple avec fopen\(\)](#)
- 2866. [Exemple avec ImageTypes](#)
- 2867. [Exemple avec le domaine par défaut](#)
- 2868. [Exemple avec mcrypt_create_iv](#)
- 2869. [Exemple avec mcrypt_get_cipher_name](#)
- 2870. [Exemple avec Soundex](#)
- 2871. [Exemple avec un formulaire simple](#)
- 2872. [Exemple avec yp_first](#)
- 2873. [Exemple complet avec lien authentifié](#)
- 2874. [Exemple d'authentification HTTP](#)
- 2875. [Exemple d'introduction](#)
- 2876. [Exemple d'ordre NIS](#)
- 2877. [Exemple d'utilisation de la fonction @ref{function.getallheaders . . getallheaders\(\) }](#)
- 2878. [Exemple d'utilisation de la fonction @ref{function.unset . . unset\(\) }](#)
- 2879. [Exemple DBA](#)
- 2880. [Exemple de création de base MySQL](#)
- 2881. [Exemple de fonction variable](#)
- 2882. [Exemple de gestion des erreurs durant création d'image \(gracieusement offert par courtesy vic@zysys.com \)](#)
- 2883. [Exemple de lecture de ligne](#)
- 2884. [Exemple de maître NIS](#)
- 2885. [Exemple de message du débogueur](#)
- 2886. [Exemple de recherche NIS](#)

- 2887. [Exemple de table de traduction](#)
- 2888. [Exemple ereg\(\)](#)
- 2889. [Exemple getrusage](#)
- 2890. [Exemple gzopen\(\)](#)
- 2891. [exemple mhash_get_hash_name](#)
- 2892. [Exemple MySQL connect](#)
- 2893. [Exemple MySQL data seek](#)
- 2894. [Exemple MySQL close](#)
- 2895. [Exemple pdfclock de la distribution pdflib 2.0](#)
- 2896. [Exemple pdfclock de la distribution pdflib 2.x](#)
- 2897. [Exemple SetCookie](#)
- 2898. [Exemple simple ClibPDF](#)
- 2899. [Exemple SWF](#)
- 2900. [Exemples](#)
- 2901. [Exemples @ref{function.setcookie . . setcookie\(\)}](#)
- 2902. [Exemples avec @ref{function.array . . array\(\)}](#)
- 2903. [Exemples avec @ref{function.array-splice . . array_splice\(\)}](#)
- 2904. [Exemples avec @ref{function.array-unshift . . array_unshift\(\)}](#)
- 2905. [Exemples avec @ref{function.array-values . . array_values\(\)}](#)
- 2906. [Exemples avec @ref{function.each . . each\(\)}](#)
- 2907. [Exemples avec @ref{function.easter-date . . easter_date\(\)}](#)
- 2908. [Exemples avec @ref{function.extract . . extract\(\)}](#)
- 2909. [Exemples de masques invalides](#)
- 2910. [Exemples de masques valides](#)
- 2911. [Expressions régulières](#)
- 2912. [Extraction d'un numéro de page d'une chaîne.](#)
- 2913. [Extraction de tous les numéros de téléphone d'un texte.](#)

F

- 2914. [Fermer une connexion Informix](#)
- 2915. [Fixer la valeur d'une variable d'environnement](#)
- 2916. [Fonctions à nombre d'arguments variable](#)
- 2917. [Fonctions calendrier](#)
- 2918. [Formulaire de chargement de fichier](#)
- 2919. [fsockopen example](#)

G

- 2920. [Générer et intercepter une erreur](#)
- 2921. [GetImageSize](#)
- 2922. [GetImageSize qui retourne IPTC](#)
- 2923. [getlastmod\(\) example](#)
- 2924. [Getting the size of the shared memory block](#)
- 2925. [GIF creation with PHP](#)

H

- 2926. [Here doc string quoting example](#)

I

- 2927. [ImageTTFText](#)
- 2928. [Inclusion d'un image GIF](#)
- 2929. [Inclusion d'une image mémoire](#)
- 2930. [Informix fetch rows](#)
- 2931. [Informix SQL fieldproperties](#)
- 2932. [Initialisation d'un tableau](#)
- 2933. [Insertion de valeurs dans la table "catalogue"](#)

L

- 2934. [Last day of next month](#)
- 2935. [Le passage du HTML au PHP](#)
- 2936. [Lecture d'un fichier ligne par ligne](#)
- 2937. [Lire les valeurs de sqlca.sqlerrd\[x\]](#)
- 2938. [Liste tous les fichiers du dossier courant](#)
- 2939. [Liste toutes les valeurs avec l'attribut "mail"](#)
- 2940. [Lit un exemple](#)

M

- 2941. [Meta Tags Example](#)
- 2942. [Migration depuis 2.0: concaténation de chaînes](#)
- 2943. [Migration depuis 2.0: valeur retournées, ancienne façon](#)
- 2944. [Migration depuis 2.0: valeur retournées, nouvelle façon](#)
- 2945. [Migration: ancienne syntaxe if..endif](#)
- 2946. [Migration: ancienne syntaxe while..endwhile](#)
- 2947. [Migration: Migration: balises start/end](#)
- 2948. [Migration: nouvelle syntaxe if..endif](#)
- 2949. [Migration: nouvelle syntaxe while..endwhile](#)
- 2950. [Migration: premières nouvelles balises start/end](#)
- 2951. [Migration: second new start/end tags](#)
- 2952. [Mise à l'échelle](#)
- 2953. [Modification d'un attribut](#)
- 2954. [Modifier l'attribut Title](#)
- 2955. [modifying Title attribute](#)
- 2956. [mysql fetch array](#)
- 2957. [mysql fetch object](#)

N

- 2958. [Nom de champs et type SQL.](#)
- 2959. [Nombre de lignes affectées](#)

O

- 2960. [OCIColumnName](#)

- 2961. [OCIColumnSize](#)
- 2962. [OCIColumnType](#)
- 2963. [OCIDefineByName](#), [OCIDefineByName](#)
- 2964. [OCIFetchStatement](#)
- 2965. [OCILogon](#)
- 2966. [OCINewDescriptor](#)
- 2967. [OCINLogon](#)
- 2968. [OCINumCols](#)
- 2969. [OCIRowCount](#)
- 2970. [OCIServerVersion](#)
- 2971. [ODBC Setoption Examples](#)
- 2972. [Output Control example](#)

P

- 2973. [Parcourir la liste des hash](#)
- 2974. [Passer en revue une base de données.](#)
- 2975. [Payflow Pro example](#)
- 2976. [Payflow Pro raw example](#)
- 2977. [Petit exemple](#)
- 2978. [pg_cmdtuples](#)
- 2979. [PHP code for accessing MyEnum](#)
- 2980. [PHP code for accessing MyInterface](#)
- 2981. [PHP code for accessing MyStruct](#)
- 2982. [PHP code for handling CORBA exceptions](#)
- 2983. [Postgres fetch object](#)
- 2984. [Postgres retourne une ligne](#)
- 2985. [PostgreSQL fetch array](#)
- 2986. [Pré remplir un formulaire PDF](#)
- 2987. [Prépare une requête, l'exécute, et affiche le résultat](#)

R

- 2988. [read_exif_data](#)
- 2989. [Reading shared memory block](#)
- 2990. [Recherche LDAP](#), [Recherche LDAP](#)
- 2991. [Rechercher la taille d'une variable dans la table des symboles](#)
- 2992. [Remplacement de plusieurs valeurs](#)

S

- 2993. [Sample IDL file](#), [Sample IDL file](#), [Sample IDL file](#), [Sample IDL file](#)
- 2994. [Sauver et Restaurer un environnement](#)
- 2995. [Sauver/Restaurer](#)
- 2996. [Shared Memory Operations Overview](#)
- 2997. [Socket example: Simple TCP/IP client](#)
- 2998. [Socket example: Simple TCP/IP server](#)
- 2999. [Some string examples](#)
- 3000. [Stocker des données sur un serveur distant](#)

3001. [Suppression d'un attribut](#)

T

3002. [Translation](#)

3003. [Traverser une base](#)

3004. [Trier plusieurs tableaux](#)

3005. [Types mysql field](#)

U

3006. [UNIX include_path](#)

3007. [users.txt](#)

3008. [Using anonymous functions as callback functions](#)

3009. [Using PHP's CURL module to fetch the PHP homepage](#)

3010. [Utilisatin des constantes FILE et LINE](#)

3011. [Utilisation de @ref{function.parse-str . . parse_str\(\)}](#)

3012. [Utilisation de l'option /e](#)

3013. [Utilisation de paquets incrémentaux](#)

3014. [Utilisation des objets de grande taille \(Large Objects\)](#)

3015. [Utiliser @ref{function.dbase-numfields . . dbase_numfields\(\)}](#)

3016. [Utiliser un REF CURSOR issue d'une commande SELECT](#)

3017. [Utiliser un REF CURSOR issue d'une procédure enregistrée.](#)

3018. [Utiliser une ressource existante](#)

V

3019. [Vérification @ref{function.gettext . . gettext\(\)}](#)

3020. [Vérification de l'existence de \\$foo dans la table des symboles](#)

3021. [Variables complexes de formulaire](#)

W

3022. [wddx_serialize_vars](#)

3023. [Windows include_path](#)

3024. [Writing to shared memory block](#)

X

3025. [XML Transtypage XML -> HTML](#)

3026. [xmltest.xml](#)

3027. [xmltest2.xml](#)

This document was generated on 27 Octobre 2000 using the [texi2html](#) translator version 1.52 (extended by davida@detron.se, hacked by dams@nexen.net for French translation). This document was generated on 27 Octobre 2000 using the [texi2html](#) translator version 1.52 (extended by davida@detron.se, hacked by dams@nexen.net for French translation).